SAPIENZA
UNIVERSITÀ DI ROMA

Enhancing Latent Alignment Methods for NLP: From Words to Concepts

Dipartimento di ingegneria Informatica, Automatica e Gestionale (DIAG)
Dottorato Nazionale in Intelligenza Artificiale (XXXVII cycle)

Andrei Stefan Bejgu
ID number 1691229

Advisor
Prof. Roberto Navigli

Academic Year 2023/2024

Thesis defended on 23 January 2025
in front of a Board of Examiners composed by:

Prof. Gabriella Pasi (chairman)

Prof. Stefania Costantini

Prof. Antonio Chella

Prof. Giovanni Semeraro

Enhancing Latent Alignment Methods for NLP: From Words to Concepts
PhD thesis. Sapienza University of Rome

© 2024 Andrei Stefan Bejgu. All rights reserved

This thesis has been typeset by LaTeX and the Sapthesis class.

Author's email: bejgu@diag.uniroma1.it

To my family,

Abstract

A central challenge in artificial intelligence is developing systems that can truly understand human language. Achieving this requires models that capture nuanced meanings, relationships, and contextual subtleties especially in specialized applications. This thesis explores strategies in Natural Language Processing (NLP) to address this challenge, beginning with foundational word representations, advancing to sentence-level representations, and then integrating both levels to enrich interpretive depth. Finally, we investigate synthetic data generation as a method to refine these representations further using Large Language Models (LLMs).

Word representations serve as the starting point, positioning words within a continuous semantic space to encode basic lexical relationships. While these embeddings effectively capture individual word associations, they fall short in representing meaning beyond isolated terms, limiting their utility in complex tasks that demand contextual comprehension. To address this gap, the thesis progresses to sentence-level embeddings, which extend beyond individual words by integrating syntactic and semantic relationships. These context-aware embeddings provide the depth necessary for semantic similarity, sentence alignment, and cross-lingual applications, enabling models to interpret richer and more nuanced meanings within larger contexts. By integrating word and sentence-level embeddings, we achieve a multi-layered representation that captures both fine-grained lexical details and broader contextual dependencies, offering a more comprehensive approach to language understanding. This layered integration sets the foundation for the final phase of the thesis: synthetic data generation, which further refines these representations by addressing data scarcity and tailoring embeddings to task-specific needs in specialized applications.

In sum, this thesis shows how each stage—from foundational word embeddings to sentence-level representations, the integration of both, and ultimately synthetic data generation—contributes to building robust and adaptable NLP models. This structured progression provides a comprehensive pathway for addressing the varied and intricate challenges of language understanding, advancing NLP toward effective language comprehension in AI systems.

Contents

Part I

Enhancing Latent Alignment Methods for NLP: From Words to Concepts

Chapter 1

Introduction

Natural Language Processing (NLP) is a key area within Artificial Intelligence (AI) that aims to enable computers to understand, interpret, and generate human language. The field has evolved significantly over the years, reflecting advances in both computational capabilities and our understanding of linguistic structure (Young et al., 2018; Otter et al., 2021; Min et al., 2023a).

At its core, NLP has always faced a fundamental challenge: representing linguistic data in a way that is both comprehensible to machines and effective for understanding the nuances of human communication (Min et al., 2023a).

Early methods for word representation, such as the *bag-of-words* (BoW) model (Harris, 1954) and *one-hot encoding*, were simple but had notable limitations. The BoW approach represents a text as a collection of word frequencies, disregarding the order and relationships between words. Similarly, one-hot encoding treats each word as an independent, sparse, high-dimensional vector, leading to inefficiencies and a failure to capture semantic relationships. These classical approaches had two key drawbacks:

1. **High dimensionality**: As vocabulary sizes grow, one-hot encoding results in extremely sparse, high-dimensional vectors. This leads to resource-intensive computations and storage inefficiencies.

2. **Inefficiency**: As vocabulary size grows, one-hot vectors become increasingly sparse and computationally inefficient. This inefficiency limits the scalability of models and hinders their application to large-scale NLP tasks.

To overcome these limitations, introducing of *distributed representations*, commonly known as *word embeddings*, marked a significant breakthrough in NLP. Unlike classical methods, embeddings represent words in dense, low-dimensional vectors that encode rich semantic information, enabling machines to better capture the meaning of language and the relationships between words.

Initial models like *Word2Vec* (Mikolov et al., 2013a) and *GloVe* (Pennington et al., 2014) are examples of *static embeddings*, where each word is assigned a fixed vector regardless of the context in which it appears. Embeddings offer several advantages over classical methods:

- **Semantic similarity**: Embeddings capture semantic relationships between words, meaning that words with similar meanings have closer vector representations. For example, "dog" and "cat" would have vectors that are close in the embedding space, while "dog" and "table" would be distant (Senel et al., 2018).
- **Compactness and efficiency**: Embeddings are dense and low-dimensional, allowing models to represent large vocabularies efficiently and capture more information in less space.

While these static embeddings improved significantly over previous approaches in different tasks (Almeida and Xexéo, 2023) like *bag-of-words* and *one-hot encoding*, they were not without limitations. One major shortcoming is that they cannot account for *polysemy* (Iacobacci et al., 2015; Pelevina et al., 2016)—the phenomenon where a single word can have different meanings depending on its context. For example, static embeddings would represent the word "bank" similarly, whether it refers to a financial institution or the side of a river, despite the apparent difference in meaning.

To address this limitation, *contextualized embeddings* were introduced with models like *BERT* (Devlin et al., 2018) and *GPT* (Brown et al., 2020b). Unlike static embeddings, these models generate different vector representations for a word based on the context in which it is used. This enables them to capture the varying meanings of polysemous words. For instance, the word "bank" in the sentences "I went to the bank to deposit money" and "I sat by the river bank" would be represented by distinct vectors, reflecting their different meanings in each context.

Contextualized embeddings excel at modeling more complex relationships, not only at the word level but also across sentences, capturing both syntactic and semantic structures that span phrases and entire texts. This ability to represent words within their specific context makes contextualized embeddings particularly effective across a wide range of NLP tasks. These tasks include **text classification** problems such as Named Entity Recognition (NER) (Li et al., 2022; Mueller et al., 2020; Tedeschi et al., 2021), **machine translation** (Clinchant et al., 2019; Guo and Nguyen, 2020), and **sentiment analysis** (Hoang et al., 2019), as well as more advanced applications like **question answering** (Yang et al., 2020), and **text summarization** (Liu and Lapata, 2019; Zhang et al., 2020). This versatility has made contextual embeddings a cornerstone in modern NLP pipelines, enabling significant improvements in task performance across diverse domains.

Despite their widespread success, embeddings trained on general-purpose corpora are intentionally designed to be broad and adaptable, enabling use across various NLP tasks. However, this generality becomes a significant limitation when applied to tasks requiring understanding unique patterns or specialized terminologies. In these cases, general-purpose embeddings often struggle to capture the task-specific distinctions that are critical for optimal performance (Tang and Yang, 2024).

To address this, adapting embeddings to meet the specific distributions and requirements of individual tasks becomes essential. In this thesis, we explore techniques such as **task-specific fine-tuning** and **embedding customization** to enhance embedding effectiveness in specialized contexts. These approaches transform general-purpose embeddings into targeted representations, significantly improving task performance where a precise understanding is required.

1.1 The Task Adaptation Gap in Embedding Representations

While general-purpose embeddings have proven effective for many broad NLP tasks, they often struggle when applied to specific task distributions, creating what we refer to as the **task adaptation gap**. This gap arises when embeddings trained on broad datasets fail to capture the nuances required for specialized tasks, where a precise understanding of terminology and linguistic structures is essential.

Embeddings trained on broad, general-purpose corpora are designed to be flexible, enabling their use across a wide range of applications. However, this versatility becomes a constraint when embeddings are applied to tasks requiring precision in capturing specialized linguistic features. Tasks such as legal domain analysis or medical report processing rely on domain-specific terminology, intricate multi-word expressions, and complex syntactic patterns that general embeddings fail to represent accurately (Min et al., 2023b; Barale et al., 2023). As a result, there is a notable performance gap between embeddings optimized for general use and those fine-tuned for task-specific distributions.

One key challenge arises from **Zipf's Law** and the uneven distribution of word frequencies, where a few terms appear frequently while many others are rare or domain-specific. General embeddings often struggle with terms like "operation," which may appear frequently but hold different meanings in varied contexts, such as medical procedures, mathematical functions, or military actions. Similarly, **multi-word expressions** and specialized terms, like "option premium" in finance or "myocardial infarction" in medicine, demand representations that capture these less frequent, nuanced terms effectively—something general embeddings typically lack.

Data scarcity further amplifies the task adaptation gap. While large general datasets are abundant, specialized data is often limited, making it difficult to train embeddings that generalize well in specific contexts. Simply fine-tuning general embeddings on small, task-specific datasets is often insufficient to overcome this challenge (Min et al., 2023b).

Addressing the task adaptation gap often requires more than simple fine-tuning on limited datasets. In many cases, a combination of techniques—such as *transfer learning*, *unsupervised adaptation*, and leveraging *synthetic data generation*—becomes necessary. Recent advancements, particularly the rise of Large Language Models (LLMs), have introduced new possibilities for synthetic data generation to augment scarce training data (Ding et al., 2024). By creating artificial examples that mirror task-specific linguistic patterns, LLMs can better align embeddings with specialized tasks, narrowing the task adaptation gap and significantly improving performance in low-resource or domain-restricted environments.

In this thesis, we will explore these challenges and propose strategies to bridge the task adaptation gap, ultimately enabling embeddings to better capture the unique characteristics of specialized tasks.

1.2 Thesis Statement and Research Objectives

This thesis focuses on narrowing the **task adaptation gap** in embedding representations by advancing methodologies that enhance performance across specific task distributions. While general-purpose embeddings have proven effective in broad NLP tasks, they often fall short when applied to specialized tasks that require a more nuanced understanding of domain-specific linguistic patterns and terminologies.

Our journey begins with improving **word-level representations**, adapting them better to capture the complexities of specialized vocabularies and multi-word expressions. From there, we expand the scope to **sentence representations**, which play a crucial role in tasks that require representing entire sentences and their underlying relationships, such as sentence alignment.

Expanding from individual word and sentence representations, this thesis underscores the power of integrating both levels to enhance nuanced understanding across various linguistic tasks. Word embeddings capture precise lexical nuances, while sentence embeddings provide a broader contextual framework. By combining these representations, we can address the need for both granular detail and overall context, proving especially valuable in tasks like information retrieval and question answering where both types of insight are critical. This integrated approach supports more accurate interpretation of specialized terminology and complex language structures, equipping models to meet the demands of sophisticated, domain-specific NLP applications.

Finally, we explore how **Large Language Models** (LLMs) can be leveraged for **synthetic data generation** in low-resource settings. By creating artificial data that mirrors the linguistic patterns of specialized tasks, LLMs offer a powerful solution to data scarcity, allowing us to augment training and further bridge the task adaptation gap. In addition to enriching datasets, synthetic data generation through LLMs facilitates the development of more precise and context-sensitive representations, aligning embeddings more closely with the unique feature of each task. This approach not only improves representation quality for complex tasks but also promotes the creation of adaptable and effective models even in domains where annotated data is limited.

Through this multi-faceted approach, the thesis aims to transform general-purpose embeddings into powerful, task-specific representations, improving performance across

various specialized NLP tasks. In summary, we lay out the following main objectives that the work presented in the thesis has aimed to accomplish the following:

- **Enhancing Word and Sentence representations for Specialized Applications**: While general-purpose embeddings perform well in broad NLP tasks, they often fall short in specialized domains, particularly in low-resource settings or when dealing with unique linguistic patterns. This thesis introduces methodologies for adapting and fine-tuning word and sentence embeddings to these challenging scenarios, leading to more accurate representations and better performance in tasks like word alignment and sentence alignment.
- **Integrating Word and Sentence Representations for Richer Semantic Understanding**: To achieve a deeper semantic understanding in complex NLP tasks, this thesis explores how combining word-level and sentence-level representations can overcome the limitations of using either type alone. It demonstrates the benefits of such integration through applications like Word Sense Linking. It shows how context from sentence embeddings and precision from word embeddings can work together to enhance disambiguation and retrieval tasks.
- **Utilizing LLMs for Synthetic Data Generation to Address Data Scarcity**: Data scarcity is a common challenge in specialized domains. This thesis investigates how Large Language Models can be used to generate synthetic data that captures domain-specific linguistic patterns. By using these synthetic datasets to supplement training data, the research shows how embeddings can be better aligned to task-specific needs, improving model performance even in low-resource settings and advancing end-to-end systems for tasks like factuality evaluation.

1.3 Publications and Contributions

The research presented in this thesis has been conducted at Sapienza NLP, the research group directed by Professor Roberto Navigli, over the three years of the National Ph.D. in Artificial Intelligence program, as part of an industrial Ph.D. conducted jointly with Babelscape.

The majority of this thesis has already been published in top-tier NLP and AI conferences. In this thesis, we discuss in-depth only those publications that are thematically close

to the aforementioned research objectives.

In what follows, we provide a reference to these publications, in chronological order:

1. Federico Martelli, **Andrei Stefan Bejgu**, Cesare Campagnano, Jaka Čibe, Rute Costa, Apolonija Gantar, Jelena Kallas, Svetla Koeva, Kristina Koppel, Simon Krek, Margit Langemets, Veronika Lipp, Sanni Nimb, Sussi Olsen, Bolette Sandford Pedersen, Valeria Quochi, Ana Salgado, László Simon, Tiberius Carole, Rafael-J Ureña-Ruiz, Roberto Navigli, 2023. XL-WA: A Gold Evaluation Benchmark for Word Alignment in 14 Language Pairs. In *Italian Conference on Computational Linguistics, CLiC-it 2023*.

   Contribution: I proposed and conducted the experimental setup, while Federico supervised the manual annotations.

2. Francesco Molfese, **Andrei Stefan Bejgu**, Simone Tedeschi, Simone Conia, Roberto Navigli, 2024. CroCoAlign: A Cross-Lingual, Context-Aware and Fully-Neural Sentence Alignment System for Long Texts. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers), EACL 2024*.

   Contribution: I proposed the main idea, designed the experimental setup, and implemented the model, while Francesco conducted the experiments. I carried out the experiments related to the MT task.

3. **Andrei Stefan Bejgu**, Edoardo Barba, Luigi Procopio, Alberte Fernández-Castro, Roberto Navigli, 2024. Word Sense Linking: Disambiguating Outside the Sandbox. In *Findings of the Association for Computational Linguistics, ACL 2024*.

   Contribution: I conducted all experiments, designed and implemented the retrieval component, and supervised the annotation process conducted by Alberte, while Edoardo developed the reader component.

4. Alessandro Scirè*, A**ndrei Stefan Bejgu***, Simone Tedeschi, Karim Ghonim, Federico Martelli, Roberto Navigli, 2024. Truth or Mirage? Towards End-To-End Factuality Evaluation with LLM-OASIS. Under review at *The Computational Linguistics Journal*.

   Contribution: Alessandro and I co-proposed the main idea. I was responsible for

creating the synthetic dataset, designing and implementing the novel retrieval component for fact verification, and conducting experiments with the LLMs. Simone and Federico managed the annotation process, Karim developed the claim extraction component, and Alessandro was responsible for the NLI component.

Finally, the above work accomplished throughout the Ph.D. has given us the opportunity to expand our ideas to other areas, resulting in additional publications that are not discussed in this thesis. We still provide here a list of these publications, as working on them has certainly influenced the direction of our work:

5. Luca Moroni, Giovanni Puccetti, Pere-Lluís Huguet Cabot, **Andrei Stefan Bejgu**, Alessio Miaschi, Edoardo Barba, Felice dell'Orletta, Andrea Esuli, Roberto Navigli, 2024. Adaptation of LLMs toward Italian Language: A systematic approach. *Under Review*.

6. Karim Ghonim, **Andrei Stefan Bejgu**, Roberto Navigli, 2024. Conceptpedia: Leveraging Traditional Approaches for Multimodal Image-Text Representation. *Under Review*

∼ This page was intentionally left blank ∼

Chapter 2

Background

2.1 Overview

This chapter provides an overview of the foundational concepts and advancements in Natural Language Processing (NLP) that serve as the basis for this thesis. Starting from classical representation methods, we trace the evolution toward neural-based models that capture complex semantic relationships. We outline how these representations contribute to increasingly nuanced and adaptive language models by examining both word- and sentence-level embeddings and developing contextualized embeddings. The chapter concludes with a discussion on the role of synthetic data generation in enhancing representation quality, particularly in domain-specific applications.

2.2 From Classical Representations to Neural Models

The evolution of Natural Language Processing (NLP) can be seen as a progression from simple, surface-level representations of linguistic data to more sophisticated, neural-based models capable of capturing deeper semantic relationships. This section outlines the major milestones in this journey, focusing on word representations, from classical methods to the introduction of neural embeddings.

2.2.1 Classical Approaches to Word Representation

Bag-of-words (BoW) and *one-hot encoding* were among the earliest methods used to represent text for machine learning tasks (Harris, 1954). Despite their simplicity, these methods have notable limitations when it comes to understanding complex linguistic phenomena.

The BoW model represents text as an unordered collection of word counts or frequencies. This approach is intuitive and easy to implement, but it disregards word order and syntactic relationships. Consider the following example:

"The cat sits on the mat."
"The mat sits on the cat."

In a BoW representation, both sentences would be treated as identical, even though their meanings are quite different. By focusing solely on word occurrences without considering the structure or context, BoW fails to capture the true semantics of a sentence.

Similarly, *one-hot encoding* represents each word as a high-dimensional, sparse vector. Assuming a vocabulary size of V, each word is represented as a vector of length V, where only one entry is set to 1, and all others are 0. For example:

"dog" $\rightarrow [0, 0, 0, 1, 0, 0, 0]$
"cat" $\rightarrow [0, 1, 0, 0, 0, 0, 0]$

In this representation, the words "dog" and "cat" are orthogonal, meaning that no semantic relationship between them is captured. One-hot encoding thus fails to model word similarities, making it difficult for algorithms to learn meaningful relationships between words. Additionally, as the size of the vocabulary grows, the vectors become sparser and more computationally expensive to process.

2.2.2 The Shift to Distributed Representations

To overcome the shortcomings of classical methods, *distributed representations*, also known as *embeddings*, were introduced. These methods map words to dense, low-dimensional vectors, where the position of each word in the vector space reflects its semantic relationship with other words. This marked a significant improvement in how machines could "understand" language.

One of the most influential models in this category is *Word2Vec*, introduced by Mikolov et al. (Mikolov et al., 2013a,b). Word2Vec learns word representations by predicting words from their context (CBOW) or by predicting the context from a word (Skip-gram). This allows the model to capture semantic relationships between words based on their distribution in large corpora. For example, the model might learn the following relationships:

"king" - "man" + "woman" = "queen"

This ability to perform arithmetic operations on word vectors while still retaining semantic meaning shows the power of distributed representations. Words with similar meanings (e.g., "dog" and "cat") are represented by vectors that are close together in the embedding space, while unrelated words (e.g., "dog" and "table") are distant.

Another popular embedding model is *GloVe* (Global Vectors for Word Representation) (Pennington et al., 2014). Unlike Word2Vec, which focuses on local context windows, GloVe is trained on global co-occurrence statistics from a corpus. This allows it to capture both local and global word relationships effectively.

2.2.3 Limitations of Static Embeddings

While distributed word embeddings like Word2Vec and GloVe marked a significant advancement over classical methods, they have inherent limitations. One of the most significant drawbacks is that these embeddings are *static*. Each word has a fixed vector representation, regardless of the context in which it appears. This presents a challenge for polysemous words—words with multiple meanings (Iacobacci et al., 2015; Pelevina et al., 2016). For instance, the word "bank" can refer to a financial institution or the side of a river. In a static embedding space, "bank" would have the same vector in both of the following contexts:

"He deposited money at the bank."
"She sat by the river bank."

In both cases, the word "bank" would be mapped to the same point in the embedding space, failing to capture the different meanings of the word based on context. This limitation is particularly problematic in tasks where a word's precise meaning is crucial, such as word sense disambiguation (Pelevina et al., 2016) and machine translation (Qi et al., 2018).

2.2.4 Contextualized Embeddings: A Leap Forward

To address the limitations of static embeddings, *contextualized embeddings* were introduced with models like *BERT* (Bidirectional Encoder Representations from Transformers) (Devlin et al., 2018) and *GPT* (Generative Pretrained Transformer) (Brown et al., 2020b). These models revolutionized NLP by generating dynamic word representations that change depending on the surrounding context.

For example, in the sentence:

"He went to the bank to deposit money."

BERT would generate a different vector for the word "bank" than it would in the sentence:

"She sat by the river bank."

This contextual sensitivity allows models like BERT to capture the multiple senses of polysemous words, making them far more effective for tasks that require a nuanced understanding of language. Contextualized embeddings are generated using deep neural networks, particularly *Transformer* architectures, which allow the model to *attend* to all words in a sentence when determining the meaning of a particular word. This enables the model to capture not just word-level relationships but also syntactic and semantic structures at the phrase, sentence, and document levels.

Overall, contextualized embeddings have dramatically improved the performance of NLP systems across a wide range of tasks, from text classification and machine translation to named entity recognition and sentiment analysis. However, as we will explore later in this thesis, even these advanced models can struggle in domain-specific applications where fine-tuning is necessary to capture specialized knowledge.

2.3 From Word to Sentence: Expanding Representation to Longer Contexts

While word embeddings have been highly effective for various NLP tasks, many applications require understanding language at a higher level of abstraction—beyond individual words. Tasks such as document classification, information retrieval, and paraphrase identification

rely on understanding the meaning of entire sentences or even paragraphs. Thus, the development of *sentence embeddings* became a critical step forward in the evolution of NLP.

Traditional methods for sentence representation, such as averaging word embeddings or using a *bag-of-words* approach, were initially employed but failed to capture the complex syntactic and semantic relationships within a sentence. For instance, averaging the word embeddings of the sentence:

"The quick brown fox jumps over the lazy dog."

would result in a representation that doesn't capture the dependencies between words like "quick" and "fox" or the relationship between "jumps" and "over." These methods treat sentences as flat, unordered collections of words, neglecting the rich information encoded in word order and grammatical structure.

2.3.1 Sentence Transformers: A Breakthrough in Sentence Representation

The introduction of *transformer-based models* revolutionized sentence embeddings. The *Transformer* architecture, first introduced by Vaswani et al. (2017), enabled models to process entire sentences in parallel, rather than sequentially. The key innovation in transformers is the *self-attention mechanism*, which allows each word to attend to all other words in a sentence, regardless of their position. This made transformers far more effective at capturing long-range dependencies and the overall meaning of sentences.

However, while highly effective at generating contextualized word embeddings, transformer-based models like BERT were not originally designed to produce sentence-level embeddings directly. BERT's default architecture is optimized for token-level tasks, such as token classification and named entity recognition, rather than sentence-level tasks. Early methods involved pooling the token embeddings (e.g., averaging or using the [CLS] token) to obtain sentence embeddings from BERT. However, these approaches often resulted in suboptimal performance for sentence-level tasks like sentence similarity or paraphrase detection (Reimers and Gurevych, 2019).

To address this limitation, *Sentence-BERT* (SBERT) (Reimers and Gurevych, 2019) was introduced as an adaptation of BERT specifically for sentence representation. SBERT fine-tunes BERT for sentence-level tasks by adding a Siamese network structure that pro-

cesses two sentences in parallel, followed by a pooling operation to generate fixed-length sentence embeddings. This architecture allows SBERT to compute sentence similarity in a highly efficient manner, making it suitable for tasks like:

> *Paraphrase Identification:* "The cat sat on the mat." $\rightarrow$ "A cat is sitting on the mat."

SBERT generates sentence embeddings that preserve the semantic relationships between sentences, enabling models to determine that these two sentences are paraphrases despite the difference in wording. SBERT has significantly improved tasks such as semantic textual similarity (STS), sentence clustering, and information retrieval.

2.3.2 Applications of Sentence Embeddings

Sentence embeddings, particularly those generated by SBERT and related models, have proven useful in many applications. In **semantic search**, for example, sentence embeddings allow search engines to retrieve results based on the meaning of a query rather than simple keyword matching. Consider the following example:

> Query: "How do I install Python on my computer?"

Using sentence embeddings, the search engine would be able to return relevant results such as:

> Result: "Guide to setting up Python on Windows and MacOS."

This capability makes sentence embeddings highly valuable for applications such as **question-answering** and **dialogue systems**, where understanding the semantic intent behind user queries is crucial.

2.3.3 Challenges and Limitations of Sentence Embeddings

Despite the success of models like SBERT, several challenges remain in sentence representation. One key limitation is that sentence embeddings trained on general-purpose datasets may struggle to generalize to domain-specific tasks (Jiang et al., 2023b). For example, a sentence embedding model trained on Wikipedia data might not perform well when applied to legal

or medical documents, where the language, structure, and terminology are vastly different. Additionally, sentence embeddings are often limited in handling sentences with complex syntactic structures or multiple clauses (Reimers and Gurevych, 2019). For example:

"Although the weather was bad, we decided to go hiking."

he relationship between the two clauses ("Although the weather was bad" and "we decided to go hiking") is essential to understanding the meaning of the sentence. Current sentence embedding models may struggle to fully capture this type of nuanced meaning, particularly in longer or more complex sentences (Wang and Izbicki, 2023).

2.3.4 Synthetic Data Generation: Enhancing Sentence Embeddings with LLMs

While sentence embeddings have become central to many NLP tasks, their effectiveness often depends on the availability of high-quality, task-specific training data. Acquiring sufficient labeled data can be challenging in many specialized fields, such as legal or medical text analysis. To address this scarcity, **synthetic data generation** has emerged as a crucial technique for supplementing existing datasets, especially in low-resource scenarios.

Traditional approaches to synthetic data generation, such as **back-translation**, create paraphrases by translating text into another language and then back into the original language (Sennrich et al., 2016). This method can produce diverse examples but often introduces errors or fails to preserve the nuances of the original text. For example, a legal sentence like "The plaintiff hereby files a motion to dismiss" might be back-translated as "The claimant submits a request to reject," which fails to maintain the exact legal terminology. Similarly, **rule-based paraphrasing** methods can generate simple variations of sentences.

The recent advances in LLMs have significantly enhanced the potential of synthetic data generation, allowing for the creation of rich, context-sensitive examples that align closely with the linguistic patterns of specific tasks (Ding et al., 2024). For instance, models like *E5-Instruct* (Wang et al., 2024b) are specifically designed to generate high-quality synthetic data tailored to improve sentence embeddings. Unlike traditional methods, these models can produce nuanced sentence pairs that reflect subtle variations in meaning, enabling more precise learning of sentence semantics. For example, E5-Instruct can generate

sentences that capture the fine distinctions between phrases like "the patient experienced mild chest pain" and "the patient reported minor chest discomfort," which is critical for tasks such as semantic similarity and information retrieval in the medical domain.

This approach augments the training process and ensures that the generated data is highly relevant to the specific needs of the task. For example, in the context of question-answering systems, LLMs can generate synthetic question-answer pairs that cover a wider variety of potential queries, improving the system's ability to understand and respond to user inputs (Schmidt et al., 2024). Similarly, for sentiment analysis, LLMs can create synthetic reviews that include different tones and sentiments, helping models to better differentiate between subtle expressions of satisfaction or dissatisfaction (Zhang et al., 2024).

By leveraging LLMs for synthetic data generation, it becomes possible to bridge the gap between general-purpose training and the specialized requirements of domain-specific sentence embeddings. This results in improved model robustness and a deeper understanding of complex linguistic nuances, making the embeddings more effective for downstream tasks like document classification, legal text analysis, and medical information extraction. Ultimately, the ability to generate tailored examples allows LLMs to support the creation of sentence representations that are better aligned with the precise demands of specialized applications (Wang et al., 2024b).

However, despite their strengths, LLMs are prone to generating **hallucinations**—outputs that are fluent and coherent but not grounded in factual data or the context provided (Tonmoy et al., 2024; Tam et al., 2022). To effectively use synthetic data generated by LLMs, it is crucial to implement mechanisms that control the quality of generated examples and to evaluate models rigorously on their ability to produce factually accurate content. This involves developing strategies to constrain the generation process and integrating evaluation metrics that assess the factual alignment of synthetic outputs. By focusing on these aspects, we can ensure that the benefits of LLM-driven synthetic data are harnessed while minimizing the risks associated with inaccurate or misleading generations.

∼ This page was intentionally left blank ∼

Chapter 3

From Classical to Contextualized Word Alignment

Abstract

This chapter examines the development of word-level representations for cross-lingual alignment tasks. First, we present statistical alignment methods, including models like GIZA++ and FastAlign, which use co-occurrence patterns and probabilistic models to estimate alignment between words across languages. These foundational approaches highlight the initial strides in word alignment and underscore their limitations. Next, we explore neural-based models that surpass statistical methods by capturing richer, context-aware word representations. By leveraging pre-trained embeddings and fine-tuned architectures, these models address the limitations of statistical methods, offering enhanced accuracy in cross-lingual word alignment. Techniques like SimAlign, which uses static and contextualized embeddings, allow for more nuanced alignment, while advanced neural architectures like MultiMirror and SQuAD-style BERT further improve alignment by incorporating task-specific training.

A key contribution of this chapter is the introduction of a benchmark covering 14 language pairs. This resource provides a broad evaluation set for cross-lingual alignment. The insights derived from word-level representations set the stage for the next chapter's focus on sentence-level representations, where broader semantic relationships and contextual nuances enhance cross-lingual understanding.

3.1 Overview

We begin our journey into the diverse landscape of representation learning by focusing on **word representations**, a foundational element in Natural Language Processing. Understanding how words are represented and aligned across languages is crucial for many cross-lingual NLP tasks, including machine translation, information retrieval, and cross-lingual annotation projection. Among these, **word alignment** plays an important role, as it involves identifying translation correspondences at word and multi-word levels between parallel sentences (Vogel et al., 1996; Tiedemann, 2004).

Historically, word alignment was central to **Statistical Machine Translation (SMT)** (Och et al., 1999; Fraser and Marcu, 2007), providing the backbone for early bilingual systems. Statistical methods, such as IBM models and HMM-based alignment approaches (Brown et al., 1993), relied on word co-occurrence frequencies and probabilistic models to estimate alignments between parallel texts. These models offered a structured framework and were instrumental in handling word correspondences across languages. However, they often struggled with more nuanced semantic relationships, particularly in polysemy and context-dependent meanings, as they lacked a deeper understanding of language beyond co-occurrence statistics.

The next major shift came with the introduction of **distributed word embeddings**, such as *Word2Vec* (Mikolov et al., 2013a) and *GloVe* (Pennington et al., 2014). These embeddings represent words as dense vectors, where the distance between vectors in the embedding space reflects semantic similarity. Unlike classical methods, word embeddings capture more subtle word relationships, moving beyond surface-level co-occurrences. This improvement made them suitable for alignment tasks, as they could recognize similarities between words that share contextual usage. For example, embeddings could position the English word "dog" close to its French counterpart "chien" in the vector space, facilitating better alignment through semantic similarity.

However, these embeddings were often **monolingual**, meaning that they were trained separately for each language, resulting in vector spaces that were not inherently aligned across languages. To apply monolingual embeddings in cross-lingual word alignment tasks, researchers developed methodologies for **mapping embeddings to a shared multilingual space**. One approach involves learning word embeddings for each language independently

and then aligning these embeddings through a mapping process to ensure that similar words in different languages have similar vector representations (Artetxe et al., 2018). This is achieved by learning a transformation that aligns the vector spaces of two languages, ensuring that words with similar meanings share similar positions across these spaces. The idea is grounded in the assumption that two equivalent words in different languages should exhibit similar distributional patterns. Thus, their vector representations should be comparable in a shared space (Artetxe et al., 2018).

In this approach, the embeddings of different languages are aligned such that they share similar geometrical shapes along the same axes, enabling vectors of semantically equivalent words across languages to be positioned closely together (Smith et al., 2017a). For instance, after mapping, the English word "dog" and the French word "chien" would be closely aligned in the shared vector space, making cross-lingual alignment more feasible.

Despite these advancements, these embeddings remained *static*, meaning that each word was assigned a single vector representation, regardless of the context in which it appeared. This static nature posed challenges for disambiguating words with multiple meanings—such as "bank," which could refer to a financial institution or the side of a river—leading to difficulties in precise word alignment. The static nature of early multilingual embeddings meant that alignment could struggle when the context was critical to understanding a word's meaning, necessitating the development of more context-sensitive alignment techniques.

The rise of **contextualized embeddings** marked another leap in the field. Models like *BERT* (Devlin et al., 2018) and other Transformer-based architectures introduced dynamic word representations that change according to their surrounding context. This advancement allowed for a deeper understanding of word meanings in different sentences, making contextualized embeddings particularly effective for word alignment. For instance, these models could differentiate between "bank" in "river bank" versus "financial bank," providing a more accurate understanding of translation correspondences. This capability significantly enhanced alignment accuracy, especially for tasks requiring disambiguation of polysemous terms or complex multi-word expressions.

A further evolution in this area came with the development of **multilingual models**, such as *mBERT* (Multilingual BERT) (Devlin et al., 2018) and *XLM-R* (XLM-Roberta)

(Conneau et al., 2020). These models are trained on large corpora across multiple languages simultaneously, learning to produce contextualized word representations aligned across languages. Unlike monolingual models that require a separate alignment process to map embeddings into a shared space, *mBERT* and *XLM-R* inherently align similar words across different languages through shared training objectives and a common vocabulary. This means that words like "dog" in English and "chien" in French are semantically similar and positioned closely in the shared embedding space generated by these models.

These multilingual models have proven particularly valuable for **zero-shot learning** scenarios, where models are applied to languages not seen during training. For example, *XLM-R* has demonstrated strong performance in zero-shot cross-lingual tasks, making it a powerful tool for word alignment in low-resource languages. The ability of these models to generalize across languages, including those with limited data, allows for more accurate alignment without requiring extensive parallel data (Jalili Sabet et al., 2020). By leveraging the multilingual nature of these models, researchers have pushed the boundaries of cross-lingual NLP, making the creation of aligned word pairs more accessible even for underrepresented languages.

With the power of contextualized models, word alignment has been revitalized in various modern NLP applications beyond traditional SMT. For instance, it has been leveraged effectively in cross-lingual annotation projection (Yarowsky and Ngai, 2001), where aligned data helps transfer linguistic annotations from high-resource to low-resource languages, creating silver datasets for a range of NLP tasks. Recent work by Procopio et al. (Procopio et al., 2021) exemplifies using neural alignment methods to create high-quality sense-tagged datasets. By leveraging contextualized embeddings for word alignment e.g. mBERT, Procopio et al. were able to achieve state-of-the-art results in cross-lingual label projection, contributing significantly to tasks like Word Sense Disambiguation (WSD) (Barba et al., 2021b; Bevilacqua et al., 2021) and Semantic Role Labeling (SRL) (Padó and Lapata, 2009; Daza and Frank, 2020). These efforts address the knowledge acquisition bottleneck (Gale et al., 1992) that is especially challenging when working with mid- and low-resource languages.

While these neural models have pushed the boundaries of word alignment, achieving increasingly better performance, they are not without limitations. One major challenge is

the **lack of high-quality manual data** in multiple languages, which significantly limits their potential and scalability. This scarcity of annotated data is particularly problematic when dealing with low-resource languages, which restricts the models' ability to generalize effectively. Addressing this issue is essential for making neural models viable across various linguistic settings.

In this chapter, we not only introduce (Martelli et al., 2023, XL-WA), a new evaluation resource for word alignment, but also go beyond this contribution by providing a detailed comparison between **classical statistical models**, **word embedding-based models**, and **neural architectures**. This comparison aims to highlight the evolution of methods in word alignment, analyzing how each approach—statistical, embedding-based, and contextual—addresses the inherent complexities of cross-lingual alignment. By examining these approaches side by side, we can better understand the trade-offs and advancements that have shaped the current state of word alignment.
Our contributions in this chapter are as follows:

1. We propose a fully manually annotated evaluation benchmark for word alignment with a total of 14 language pairs, each composed of English and one of the following languages: Arabic, Bulgarian, Chinese, Danish, Dutch, Estonian, Hungarian, Italian, Korean, Portuguese, Russian, Slovenian, Spanish, and Swedish.

2. We conduct experiments focusing on word-level representations using both statistical and neural approaches to word alignment. By evaluating each method's performance on our newly created benchmark, we aim to provide a thorough comparison of their capabilities at the word level.

3.2 Related Work

Approaches Initial approaches to word alignment leveraged statistical and heuristic models (Della Pietra, 1994). Along these lines, several systems were proposed such as HMM (Vogel et al., 1996), GIZA++[1] (Och and Ney, 2003), PGIZA++, MGIZA++ (Gao and Vogel, 2008), and FastAlign[2] (Dyer et al., 2013). These methods were based on the statistical

[1] https://github.com/moses-smt/giza-pp
[2] https://github.com/clab/fast_align

relationships between words in parallel corpora, relying heavily on word co-occurrence frequencies and probabilistic modeling to estimate alignment. These models were effective in the era of Statistical Machine Translation (SMT), where they played a central role in building alignment tables used for translation. However, they often struggled with handling polysemy and failed to fully capture the semantic nuances of word relationships due to their reliance on surface-level co-occurrence statistics.

Embedding-Based Methods With the rise of distributed word embeddings, new possibilities emerged for word alignment by leveraging semantic similarity between word vectors across languages. If similar words in different languages have similar embeddings, these embeddings can be used to compute a **similarity matrix** between parallel sentences, allowing alignments to be identified without requiring parallel training data. This concept underlies the **SimAlign** method (Jalili Sabet et al., 2020), which constructs a similarity matrix between the embeddings of words from two parallel sentences. SimAlign can work with embeddings generated from pre-trained multilingual models such as *mBERT* (Devlin et al., 2018) and *XLM-R* (Conneau et al., 2020), enabling it to align words across languages without needing further training.

SimAlign offers several alignment strategies, including **Argmax**, **Itermax**, and **Match**. The Argmax method optimizes local alignment by aligning each word with its most similar counterpart in the other language. However, mutual argmax alignments can be sparse, so SimAlign also provides the Itermax method, which iteratively identifies alignment pairs, focusing on unaligned words while allowing for one-to-many alignments. The Match method, in contrast, finds global alignment optima, ensuring bidirectional alignment consistency between the source and target words. These approaches make it possible to directly apply multilingual embeddings—from static word embeddings aligned into a shared vector space or directly from multilingual models like mBERT and XLM-R—for word alignment tasks.

Neural Approaches More sophisticated neural models for word alignment were developed. These models utilize the powerful representation capabilities of transformers to achieve state-of-the-art performance in alignment tasks. Some examples include the work of Garg et al. (2019), who proposed a neural joint alignment model, and Zenkel et al. (2019), who enhanced

alignment performance by incorporating explicit alignment layers into Transformer-based translation models. Stengel-Eskin et al. (2019) and Nagata et al. (2020) further explored supervised neural approaches that leverage labeled alignment data for improved accuracy.

More recently, Procopio et al. (Procopio et al., 2021) introduced a neural discriminative model for word alignment based on multilingual BERT (Devlin et al., 2019). This approach takes advantage of the pre-trained knowledge in mBERT and fine-tunes it specifically for alignment tasks, achieving significant accuracy and processing time improvements. The model uses the inherent multilingualism of mBERT to align words between languages, reducing the need for extensive parallel training data and enabling effective zero-shot performance on low-resource languages. These recent advancements illustrate the continued evolution of word alignment from statistical methods to modern, context-aware neural models that leverage multilingual pre-training.

Data For the last few decades, several datasets for word alignment, both manual and automatic, have been created, e.g. Czech-English[3] (Mareček, 2008), Dutch-English (Macken, 2010), English-Turkish (Cakmak et al., 2012), English-Swedish[4] (Holmqvist and Ahrenberg, 2011), Chinese-English[5] (Liu and Sun, 2015). Interestingly, Graca et al. (Graca et al., 2008) proposed a collection of small datasets for word alignment in 6 language combinations; each dataset is composed of 100 sentences derived from the Europarl corpus[6] (Koehn, 2005). Among the currently available resources, we highlight the following contributions which we use in our experiments: the English-French and Romanian-English corpora released during the HLT-NAACL-2003 workshop on Building and Using Parallel Texts[7] (Mihalcea and Pedersen, 2003), and the German-English dataset[8] proposed by Vilar et al. (Vilar et al., 2006). Finally, Neubig (Neubig, 2011) presented a Japanese-English dataset[9] obtained by translating Wikipedia pages. However, despite the preceding efforts undertaken in this direction, to the best of our knowledge, no entirely manually curated evaluation benchmark

[3] https://ufal.mff.cuni.cz/czech-english-manual-word-alignment

[4] https://www.ida.liu.se/divisions/hcs/nlplab/resources/ges/

[5] https://nlp.csai.tsinghua.edu.cn/~ly/systems/TsinghuaAligner/TsinghuaAligner.html

[6] https://www.statmt.org/europarl/

[7] https://web.eecs.umich.edu/~mihalcea/wpt/

[8] https://www-i6.informatik.rwth-aachen.de/goldAlignment/

[9] http://www.phontron.com/kftt

	# of Sentences		# of Alignments	
Lang.	**Dev**	**Test**	**Dev**	**Test**
EN-AR	90	210	1591	3597
EN-BG	105	245	1719	4179
EN-DA	105	245	1841	4136
EN-ES	105	245	1961	4722
EN-ET	105	245	1614	3722
EN-HU	105	245	1580	3781
EN-IT	103	243	1980	4765
EN-KO	90	210	1277	3007
EN-NL	105	245	1886	4490
EN-PT	105	245	1849	4578
EN-RU	90	210	1114	2582
EN-SL	105	245	1942	4537
EN-SV	90	210	1522	3530
EN-ZH	90	210	1724	4135
$\sum$	1393	3253	23600	55761

Table 3.1. Composition of XL-WA. We report from left to right: the available language combinations, the number of sentences and alignments divided by data split. In our experiments, we use approximately 30% of our data for development so as to obtain a more representative set.

that matches XL-WA in both size and language pairs covered is currently available.

3.3 XL-WA

We introduce XL-WA, a novel, entirely manually-curated evaluation benchmark for word alignment to tackle the aforementioned gap. XL-WA is currently composed of 14 datasets, of which 9 are parallel. The languages included in XL-WA cover 7 different language families, i.e. Afro-Asiatic, Indo-European (Germanic), Indo-European (Romance), Indo-European (Slavic), Sino-Tibetan, Uralic (Finno-Ugric) and Koreanic.

Importantly, all datasets include English as source language. This choice is motivated by enabling word alignment from English to multiple target languages, which is crucial for tasks such as label projection, where the majority of high-quality annotated data whose labels can be propagated is typically available in English.

We show the composition of our dataset in Table 3.1. Importantly, XL-WA is annotated exclusively by professional mother tongue annotators with a solid academic background and proven experience in linguistic annotation tasks. A detailed description of the format that we adopt is provided in Section 3.3.2.

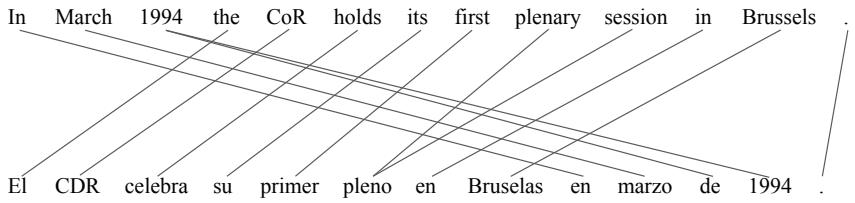

Figure 3.1. An example of alignment between English and Spanish, derived from the EN-ES dataset in XL-WA.

3.3.1 Creation process

This section details the creation process and illustrates the guidelines adopted during the annotation phase.

The creation of XL-WA can be divided into three steps: i) *automatic extraction* of candidate sentences from a corpus, ii) *manual selection* of sentences satisfying specific linguistic criteria, and iii) *manual alignment*.

To obtain a balanced corpus in terms of domains and genres, similarly to the procedure adopted by Martelli et al. (2021), we extract our data from WikiMatrix[10] (Schwenk et al., 2021), a wide-coverage collection of parallel sentences derived from the Wikipedia[11] corpus using an automatic approach based on multilingual sentence embeddings, covering 1620 different language combinations. First, we consider the WikiMatrix datasets containing English as the source language, and extract the highest number of overlapping source sentences across datasets. To this end, we compute a Boolean matrix $A \in \{0,1\}^{m \times n}$ where m is the number of English sentences in WikiMatrix and n the number of the target languages other than English covered in XL-WA.

We compute A such that A_{ij} contains: i) 1 if, for the i-th English sentence, a translation into the j-th target language is available, ii) 0 otherwise. We first extract the sentences shared in the highest number of languages. Subsequently, we manually discard

[10] https://ai.facebook.com/blog/wikimatrix/

[11] https://www.wikipedia.org/

sentences that are not well-formatted or contain significant grammatical errors. We then ask annotators to provide the missing translations to fill the gaps in our parallel dataset[12]. Finally, we ask our annotators to perform word alignment from scratch.

Guidelines All annotators are required to follow specific annotation guidelines for word alignment inspired by Lambert et al. (Lambert et al., 2005), who provide detailed instructions and suggestions regarding the annotation of datasets for word alignment, including specific cases and exceptions. Importantly, annotators are asked to align source and target words also when these do not share the same part of speech. Furthermore, annotators must align complex lexical units such as compounds and multi-word expressions. For instance, given an open compound word c_{en}, e.g. *bus driver* in the English source sentence, translated into Dutch with the compound word c_{nl} *buschauffeur*, each component of c_{en} should be aligned to c_{nl}.

3.3.2 Alignment Format

We now describe our alignment format and provide an example for the language combination English-Spanish (EN-ES).

We adopt the Pharaoh alignment format (Koehn, 2004). Specifically, we use a Tab-Separated Values (TSV) format, where each row is formatted as follows: `source sentence<tab>target sentence<tab>alignments`. Spaces separate tokens and alignments; each alignment is composed of a pair of integers which identify the corresponding positions of source and target tokens, starting from zero. In order to deal with multi-word expressions in which 1:1 alignments are not possible, e.g., due to collocations or idiomatic expressions, we align all components of a given multi-word expression in English with all components of the corresponding multi-word expression in the target language.

Below, we report an example extracted from the EN-ES dataset:

- **Source:** `In March 1994 the CoR holds its first plenary session in Brussels .`

[12]Due to time constraints and the limited availability of professional annotators for specific language combinations, we carry out this step in 9 language combinations only.

- **Target:** `El CDR celebra su primer pleno en Bruselas en marzo de 1994 .`
- **Alignments:** `3-0 4-1 5-2 6-3 7-4 8-5 9-5 10-6 11-7 0-8 1-9 2-10 2-11 12-12`

A visual representation of the above example is provided in Figure 3.1.

3.3.3 Inter-annotator agreement

Finally, in order to assess the reliability of our manual annotations, we compute the inter-annotator agreement[13]. To this end, we randomly select a sample of approximately 50 sentence pairs in two language combinations, namely EN-DA and EN-IT, and ask new annotators to align these manually. We compute the Cohen's kappa and obtain 0.94 and 0.89 in EN-DA and EN-IT, respectively. Importantly, these results indicate a remarkable level of agreement, which suggests a high degree of annotation consistency across datasets.

3.4 Experimental Setup

In this section, we illustrate our experimental setup and carry out a performance analysis. To this end, we put forward two different experimental settings. Specifically, we propose a comparison between statistical and neural approaches tested against our novel benchmark in a language-specific setting, i.e. we train and test on the same language pairs (Section 3.4.1).

3.4.1 Settings

Language-specific setting

Systems In this setting, we experiment with various word alignment methods, progressing from statistical models to more advanced neural-based approaches.

We begin with two classical statistical methods, **GIZA++** (Och and Ney, 2003) and **FastAlign** (Dyer et al., 2013). These models rely on probabilistic alignment models, utilizing word co-occurrence statistics in parallel corpora to generate alignments. GIZA++ uses iterative expectation maximization to refine word-to-word alignments over multiple passes,

[13] Due to time constraints, we compute the inter-annotator agreement in two language combinations.

while FastAlign simplifies the alignment process with a more efficient, fast-asymmetric model.

Next, we explore the use of **static word embeddings** for word alignment through **parallel FastText embeddings** (Bojanowski et al., 2017). The parallel embeddings are generated using the method proposed by Smith et al. (2017b), which aligns monolingual FastText embeddings into a shared bilingual vector space through orthogonal transformations. To derive alignments, we use the **SimAlign** method, which constructs a similarity matrix between the embeddings of words from parallel sentences. From this matrix, SimAlign identifies word pairs with high similarity scores, using the **itermax** technique to determine the most probable alignments. This approach allows us to align words without the need for additional training data.

Building upon this, we apply the same SimAlign methodology using **contextualized embeddings** generated by *multilingual BERT (mBERT)* and *XLM-R*. Unlike static embeddings, mBERT and XLM-R produce dynamic word representations that adapt to the context in which a word appears, providing richer semantic information for alignment. SimAlign leverages these contextualized embeddings to create similarity matrices, enabling the alignment of words across languages with a more nuanced understanding of context. Using the same SimAlign method for static and contextualized embeddings, we can directly compare the impact of context-aware representations on alignment quality.

Finally, we evaluate two state-of-the-art **neural models** for word alignment that are explicitly trained for this task. The first is the SQuAD-style formulation for word alignment proposed by Nagata et al. (Nagata et al., 2020), which fine-tunes mBERT to align words by framing the problem as a question-answering task. The second is the **MultiMirror neural word aligner** by Procopio et al. (Procopio et al., 2021), which uses a discriminative neural approach based on mBERT, optimizing the alignment process for accuracy and processing speed. These neural models are designed to leverage the pre-trained multilingual knowledge of mBERT, adapting it specifically for the word alignment task through supervised training, thus offering a more targeted solution than SimAlign.

Data For this setting, we derive training data from three well-established parallel corpora, namely, **Europarl**, **WikiMatrix**, and **UNPC**[14] (Ziemski et al., 2016). These parallel corpora provide a diverse and comprehensive source of sentence pairs, covering all language combinations considered in our experiments. For each language pair, the statistical alignment systems (GIZA++ and FastAlign) are trained on a randomly selected sample of 0.5 million parallel sentences from these corpora, concatenated with our test data. This ensures that the statistical models can access a sufficient quantity of parallel data to learn reliable word co-occurrence patterns.

For the neural models, which require aligned data for training, we follow the approach proposed by Garg et al. (Garg et al., 2019) due to the absence of gold-standard aligned data across all language pairs. Specifically, we use silver-standard training data derived from alignments produced by the statistical models. This process involves tagging sentences with alignments from GIZA++ and FastAlign and selecting 1,000 sentences per language pair with the highest alignment overlap. By using this high-confidence subset of aligned sentences, the neural models can better learn to adapt pre-trained multilingual embeddings, such as those from mBERT, to the specific requirements of word alignment.

For validation and evaluation, we utilize the **XL-WA** datasets, whose composition is detailed in Table 3.1. These manually curated datasets enable us to assess the performance of each model on different language pairs, providing a benchmark for evaluating the effectiveness of each method in capturing translation correspondences.

3.4.2 Evaluation metrics

As customary in the word alignment task, we adopt the F1 metric. In this work, we do not use the Alignment Error Rate (AER) metric since previous works argue that AER is unlikely to be a useful metric for word alignment due to its bias towards precision (Fraser and Marcu, 2007).

	Statistical		SimAlign (Static/Contextualized)			Neural (Trained)	
	GIZA++ (Och and Ney, 2003)	FastAlign (Dyer et al., 2013)	SimAlign FastText (Bojanowski et al., 2017)	SimAlign mBERT (Devlin et al., 2018)	SimAlign XLM-R (Conneau et al., 2020)	SQuAD BERT (Nagata et al., 2020)	MultiMirror (Procopio et al., 2021)
EN-AR	61.6	67.4	71.1	74.2	75.2	82.9	82.8
EN-BG	74.1	70.4	74.3	82.1	83.3	85.8	86.9
EN-DA	74.3	72.0	78.3	88.3	89.2	92.3	92.1
EN-ES	74.8	73.5	78.2	83.5	84.3	87.6	86.8
EN-ET	63.3	64.4	68.6	76.2	77.4	81.0	81.7
EN-HU	59.6	57.7	63.9	73.4	74.8	76.6	76.1
EN-IT	52.9	54.6	58.2	80.3	81.1	84.0	83.2
EN-KO	52.2	51.7	54.7	68.0	68.6	42.1	69.9
EN-NL	79.0	78.0	81.8	91.7	92.3	94.3	93.8
EN-PT	76.7	75.1	76.7	86.2	86.7	88.8	87.9
EN-RU	73.8	74.5	77.4	83.6	84.3	84.9	85.8
EN-SL	68.6	67.9	70.4	80.1	81.2	82.4	83.3
EN-SV	75.8	73.9	77.0	89.0	89.6	89.2	89.3
EN-ZH	44.5	48.9	51.9	72.0	72.9	74.5	75.3
Avg	66.5	66.4	71.3	80.9	82.0	81.9	83.9

Table 3.2. Comparison between statistical methods (GIZA++ and FastAlign), SimAlign-based methods using static and contextualized embeddings (FastText, mBERT, XLM-R), and current state-of-the-art neural approaches (SQuAD BERT and MultiMirror). All values represent the F1-score for each language pair.

3.5 Results

In this section, we discuss the results obtained across different word alignment methods. As shown in Table 3.2, there is a clear performance gradient among the various approaches, with statistical methods, SimAlign-based approaches, and neural models forming distinct tiers of effectiveness. Overall, the neural methods significantly outperform the statistical approaches, achieving improvements of up to 17.5 points in terms of F1 score on average.

Statistical vs. Neural Approaches The statistical methods, namely **GIZA++** and **FastAlign**, serve as baselines in this evaluation. While these approaches provide reasonable performance, their reliance on word co-occurrence statistics limits their ability to handle the complexity of modern word alignment tasks, especially in the face of polysemy and nuanced semantic variations across languages. For example, in English-Dutch (EN-NL), GIZA++ achieves an F1 score of 79.0, whereas neural models like **SQuAD BERT** reach an F1 score of 94.3, highlighting the gap between traditional and modern methods.

[14] https://opus.nlpl.eu/UNPC.php

SimAlign with Static and Contextualized Embeddings SimAlign bridges the gap between purely statistical methods and more advanced neural models by leveraging pre-trained embeddings to perform alignment. Using static word embeddings, such as **FastText** (Bojanowski et al., 2017), SimAlign shows an improvement over the statistical methods, with an average F1 score of 71.3. This suggests that even without specialized training, using semantic similarity between pre-trained embeddings can significantly enhance alignment quality.

The results further improve when using contextualized embeddings with SimAlign, such as those derived from **mBERT** (Devlin et al., 2018) and **XLM-R** (Conneau et al., 2020). For instance, SimAlign with mBERT achieves an average F1 score of 80.9, while SimAlign with XLM-R reaches 82.0. These models benefit from the ability to generate context-specific representations of words, making them particularly useful for distinguishing between polysemous words (e.g., "bank" in "river bank" versus "financial bank"). The improvements over static embeddings demonstrate the value of context awareness in word alignment tasks.

Neural Approaches The best performance across all methods is obtained using fully trained neural models, specifically the **SQuAD-style BERT** formulation (Nagata et al., 2020) and the **MultiMirror** approach by Procopio et al. (Procopio et al., 2021). These models leverage supervised learning and task-specific training to fine-tune pre-trained language models for word alignment, achieving average F1 scores of 81.9 and 83.9, respectively. For example, the EN-NL combination sees SQuAD BERT reaching an F1 score of 94.3, showcasing the strength of specialized training in capturing nuanced translation correspondences.

Challenges with Low-Resource and Topic-Prominent Languages Despite their overall superiority, neural models and even SimAlign with contextualized embeddings face challenges with certain language types, such as topic-prominent languages like Chinese (EN-ZH), Hungarian (EN-HU), and Korean (EN-KO). The alignment models—including neural methods—achieve significantly lower F1 scores in these languages than other language pairs. For instance, SQuAD BERT achieves an F1 score of just 42.1 in EN-KO, reflecting the complexity of aligning languages with significant structural and typological differences from

	MultiMirror 1k sentences			MultiMirror 10k sentences		
Lang.	**P**	**R**	**F1**	**P**	**R**	**F1**
EN-AR	88.3	77.9	82.8	88.9	79.3	83.8
EN-BG	85.5	88.3	86.9	84.7	88.5	86.6
EN-DA	90.8	93.4	92.1	91.3	92.0	91.7
EN-ES	89.5	84.4	86.8	91.8	82.6	86.9
EN-ET	77.2	86.8	81.7	78.0	84.8	81.3
EN-HU	72.4	80.1	76.1	74.2	79.6	76.8
EN-IT	88.2	78.7	83.2	89.5	79.7	84.3
EN-KO	69.5	70.4	69.9	71.1	72.2	71.6
EN-NL	94.2	93.4	93.8	95.9	92.5	94.2
EN-PT	87.9	87.9	87.9	89.0	86.8	87.9
EN-RU	87.6	84.0	85.8	87.5	85.2	86.3
EN-SL	85.4	81.4	83.3	84.4	81.3	82.8
EN-SV	91.5	87.2	89.3	92.3	85.9	89.0
EN-ZH	79.2	71.7	75.3	80.5	72.3	76.2
Avg	84.8	83.3	83.9	85.7	83.0	84.2

Table 3.3. Comparison between MultiMirror trained on different size silver datasets. P, R and F1 stand for Precision, Recall and F1-score, respectively; all the scores are calculated using the Micro average.

English. Even the SimAlign-based approaches, which can capture some context through embeddings, struggle to close this gap in these cases.

These results highlight that while neural models and contextualized embeddings represent a significant advancement in word alignment, there remains room for improvement, particularly in handling low-resource and typologically diverse languages. The gap between statistical and neural models underscores the importance of context and supervised training. Still, it also emphasizes the need for further research into methods that can generalize better across diverse linguistic structures.

Training size impact on Neural approaches Finally, we investigate the impact of the size of the training data generated with GIZA++ and FastAlign, as described in Section 3.4.1, on the overall performance achieved by MultiMirror[15]. To this end, we increase the size of the silver training data to 10,000 sentence pairs and compare the results obtained with those achieved in the previous setting where we use 1,000 sentence pairs. As can be seen in Table

[15]We use MultiMirror in this experiment due to a satisfactory trade-off between performance and processing speed.

3.3, the greater quantity of data allows us to achieve better results in terms of both precision and F1 score. However, interestingly, when training on 10,000 sentence pairs, MultiMirror reports a slightly inferior performance in terms of recall, with a decrease of 0.3 on average.

3.6 Conclusions

In this chapter, we evaluated various word alignment methods, offering insights into the capabilities and limitations of different representation approaches in cross-lingual settings. Our experiments spanned classical statistical models like GIZA++ and FastAlign, as well as methods leveraging pre-trained embeddings with SimAlign, and advanced neural models specifically fine-tuned for word alignment tasks.

The findings show that while classical statistical methods maintain their utility due to their simplicity and efficiency, their performance is constrained by a reliance on surface-level co-occurrence patterns. This limitation often makes handling linguistic nuances such as polysemy and context-dependent meanings challenging. Methods using static and contextualized embeddings, such as SimAlign with FastText, mBERT, and XLM-R, offer a middle ground, leveraging richer semantic information without the need for task-specific training data. These approaches show how pre-existing representations can be adapted to new tasks, providing a more flexible but still somewhat constrained solution.

The most significant improvements were observed with neural models that undergo targeted training, such as the SQuAD-style BERT and MultiMirror approaches. These methods illustrate the advantages of adapting pre-trained models to specific tasks through fine-tuning, capturing more intricate semantic patterns and achieving higher overall accuracy. However, even these advanced models face challenges when dealing with typologically diverse languages and domain-specific linguistic structures, indicating the need for further adaptation and specialization.

Overall, our experiments highlight a key theme in representation learning: the balance between general-purpose representations with broad applicability and specialized models requiring fine-tuning to achieve peak performance in specific contexts. This balance is central to the broader landscape of NLP, where the choice of representation—whether at the word, sentence, or concept level—significantly impacts the effectiveness of downstream

applications.

In the next chapter, we broaden our exploration from word-level representations to the richer and more complex domain of sentence-level representations. By shifting our focus to sentence alignment, we aim to understand how capturing relationships between entire sentences can provide deeper insights into cross-lingual semantics. This transition highlights the evolution from representing individual words to modeling the meaning of whole sentences, addressing the challenges of maintaining contextual integrity and understanding nuanced semantic relationships across different languages. Building on the foundations laid by our analysis of word-level approaches, we now delve into how sentence embeddings can enhance tasks like cross-lingual sentence alignment, contributing to a more comprehensive understanding of multilingual text.

∼ This page was intentionally left blank ∼

Chapter 4

From Word Embeddings to Sentence Representation

Abstract

Following the discussion on word-level representations, this chapter shifts focus to sentence-level embeddings, which provide a holistic view essential for tasks like semantic similarity and sentence alignment. Initially, we examine traditional methods that use averaged word embeddings, establishing a baseline for capturing sentence meaning. While simple, these methods overlook word order and syntactic nuances, often leading to inadequate representations for complex sentence-alignment tasks.

Building on this, we present modern, transformer-based approaches like Sentence-BERT (SBERT), which capture contextual dependencies across entire sentences and address limitations such as word order sensitivity and contextual awareness. By refining BERT with a Siamese network structure, SBERT offers enhanced sentence embeddings that support nuanced tasks like cross-lingual sentence alignment, outperforming previous word-level methods and simpler sentence embeddings.

We highlight how these advanced methods improve alignment accuracy through experiments, particularly in multilingual contexts where sentence structures and nuances vary widely. This chapter concludes by setting up our transition to tasks where both word- and sentence-level representations converge to enrich deeper semantic understanding.

4.1 Overview

In the previous chapter, we explored how word-level representations can be effectively used for tasks such as word alignment, providing insights into the strengths and limitations of both statistical models and neural approaches. While word level representations have been crucial for many NLP tasks, they are often insufficient for applications requiring a more holistic understanding of language. Sentence-level representations, which capture the meaning of entire sentences rather than just individual words, are essential for handling more complex tasks that involve deeper semantic understanding.

This chapter shifts the focus from word embeddings to sentence-level representations, which have become increasingly important in tasks like **semantic similarity**, **information retrieval**, and, notably, **sentence alignment**. Sentence alignment involves matching corresponding sentences between parallel documents, such as translations or other related pairs (Abdul-Rauf et al., 2012). It plays a key role in constructing parallel corpora, which are fundamental for training models in **machine translation** (Shi et al., 2021), **text simplification** (Jiang et al., 2020), and **paraphrase generation** (Barzilay and Lee, 2003).

Unlike word alignment, where statistical methods like **GIZA++** and **FastAlign** can efficiently model word co-occurrences, applying such methods directly to sentences poses significant challenges. The primary issue is that the vocabulary space for sentences is orders of magnitude larger than that for individual words, making statistical models impractical for this task. For example, the number of possible sentence pairs in parallel corpora grows exponentially, leading to sparse data problems and high computational costs. This limitation necessitates a shift towards more sophisticated representation models that can handle the complexity of sentence-level semantics.

To address these challenges, sentence embeddings have emerged as a promising solution, allowing for the representation of sentences as dense vectors that capture their overall meaning. Early approaches to sentence representation involved averaging static word embeddings, such as those produced by *Word2Vec* (Mikolov et al., 2013a) or *GloVe* (Pennington et al., 2014). In these methods, a sentence embedding is obtained by taking the mean of the word vectors within a sentence, producing a fixed-length vector that represents the sentence's content. For example:

"The cat chased the mouse."

In this approach, the embeddings of each word (e.g., "cat," "chased," "mouse") are averaged together to create a single vector representing the entire sentence. While simple and computationally efficient, these mean-pooling methods have notable limitations. They often fail to capture word order and syntactic structures, resulting in representations that overlook the nuanced relationships between words in a sentence. For instance, averaging word embeddings treats sentences as bags of words, ignoring the fact that "The cat chased the mouse" and "The mouse chased the cat" have different meanings despite containing the same words.

While models like *BERT* (Devlin et al., 2018) have significantly advanced the ability to capture contextual word meanings, they encounter challenges when used directly for generating sentence embeddings. Research by Ethayarajh (2019) highlighted that BERT's representations tend to suffer from issues like *anisotropy*, where the sentence representations are densely clustered into a narrow cone, making most vectors appear unnaturally close to one another. This characteristic diminishes the ability to distinguish between semantically different sentences effectively. Moreover, the default use of BERT for sentence similarity tasks can be inefficient; retrieving the best match for a query can become computationally intensive and time-consuming, making it less practical for real-world applications where speed is crucial (Reimers and Gurevych, 2019).

Recognizing these limitations, more advanced models such as *Sentence-BERT (SBERT)* (Reimers and Gurevych, 2019) were introduced to address the shortcomings of these approaches. SBERT builds on the transformer-based *BERT* (Devlin et al., 2018) architecture, specifically adapting it for generating sentence embeddings. Instead of relying on simple averaging, SBERT fine-tunes BERT with a Siamese network structure, which allows it to directly compute the similarity between sentence pairs by learning embeddings that are optimized for semantic similarity tasks.

SBERT's approach to sentence representation provides several advantages. It generates embeddings that are sensitive to the context of the entire sentence, enabling more accurate comparisons between sentences in tasks such as **sentence alignment**. This allows SBERT to capture the differences between semantically distinct sentences that contain similar words but convey entirely different meanings, such as:

"The doctor treated the patient with kindness." vs. *"The patient treated the doctor with kindness."*

In this example, SBERT can produce sentence embeddings that recognize the significant difference in meaning between the two sentences, even though they contain the same words. The word order shifts the roles of "doctor" and "patient," leading to entirely different interpretations. This makes SBERT significantly improved over earlier methods that rely solely on word-level representations, allowing for more effective alignment in complex cross-lingual scenarios.

Despite advancements, many existing sentence alignment methods rely on heuristic-based approaches or use auxiliary machine translation systems to aid in alignment (Sennrich and Volk, 2011; Thompson and Koehn, 2019). These methods often neglect the role of broader context, which is critical for disambiguating the meaning of sentences. Sentences can be ambiguous when considered in isolation, especially in longer texts like books, where understanding the surrounding context is essential for accurate alignment (Gaballo, 2012).

In this chapter, we address these issues by focusing on advanced methods for sentence alignment that leverage contextualized embeddings and neural models capable of modeling broader context we introduce (Molfese et al., 2024, CroCoAlign).

To address these challenges, in this chapter, we put forward the following contributions:

- We provide a detailed analysis of different sentence representation methods, ranging from traditional approaches to advanced models like *Sentence-BERT* and other transformer-based techniques. This analysis focuses on their ability to capture context and semantic nuances that are critical for sentence alignment.

- We conduct a deep evaluation of various sentence alignment methods, comparing heuristic-based approaches with neural models that utilize contextualized embeddings. Our experiments reveal the impact of context-aware representations on alignment quality, especially in handling complex alignment scenarios like many-to-one and one-to-many alignments.

- Using a multilingual dataset derived from the *Opus Books project*[1], which covers a

[1] `https://opus.nlpl.eu/Books.php`

wide range of languages and genres, we show how different alignment techniques perform across various text types. This allows us to assess the generalizability of sentence alignment models beyond standard benchmarks.

- We highlight the challenges of aligning sentences in context-dependent situations, emphasizing the importance of advanced models that can capture broader context. Our findings show how neural models can outperform traditional methods in these scenarios, providing valuable insights for improving cross-lingual semantic tasks.

4.2 Related Work

In this section, we review the literature of the sentence alignment task. To highlight the unique aspects of this task, we also outline the differences between sentence alignment and bitext mining, a closely related task, and describe why bitext mining systems are not suitable for sentence alignment. Finally, we showcase applications of sentence alignment systems in MT, underlining the importance of the task in real-world scenarios.

4.2.1 Sentence Alignment

Traditional ways of aligning sentences were to leverage sentence length information, or to look for lexical patterns. The first sentence alignment systems relied solely on the number of words or characters within each sentence (Brown et al., 1991; Gale and Church, 1993). Similarly, Kay and Röscheisen (1993) presented an alignment algorithm based on word correspondences, while Chen (1993) calculated the probability of an alignment by using a word-to-word translation model.

To speed up computation, later research merged word-level features with sentence-level translations (Moore, 2002; Varga et al., 2007). It is also possible to align sentences based on their degree of textual and metatextual structure. For instance, Tiedemann (2007) indicated that movie subtitles can be highly attractive for alignment thanks to their time stamps. MT-based methods were introduced in subsequent literature (Sennrich and Volk, 2011), followed five years later by pruned phrase tables from a statistical MT system (Gomes and Lopes, 2016). In both the foregoing methods, high-probability one-to-one alignments were anchored in the search space and then the alignments were filled in and refined.

More recently, Thompson and Koehn (2019) introduced Vecalign, which uses a dynamic programming algorithm based on a combination of LASER (Artetxe and Schwenk, 2019) sentence embeddings and Fast Dynamic Time-Warping, setting a new state of the art in sentence alignment. Although previous work has greatly improved the performance of sentence alignment systems, we still lack an in-depth investigation on how to encode cross-sentence context in long texts.

4.2.2 Bitext Mining

Bitext mining, also known as bitext retrieval, is the task of mining sentence pairs that are translations of each other from large text corpora. Differently from sentence alignment, where global context and sequentiality play a key role and many-to-many alignments are possible, bitext mining systems focus on standalone, 1-to-1 sentence pairs. Typically, bitext mining systems undergo assessment through established benchmarks, such as the United Nations (Ziemski et al., 2016, UN), BUCC (Zweigenbaum et al., 2017), and the Tatoeba corpora (Artetxe and Schwenk, 2019). Nevertheless, these datasets are organized in a manner where, given two monolingual corpora, only a portion of them is assumed to be parallel. This suggests that the source domain can vary greatly from one sentence to another, thereby being in significant contrast with sentence alignment datasets, where the domain tends to remain consistent throughout the entire document. For this reason, state-of-the-art bitext mining systems, such as LASER (Artetxe and Schwenk, 2019) and LaBSE (Feng et al., 2022), are not designed to handle sequential relationships between sentences within a document, and overlook situations where source or target sentences are fragmented into multiple segments.

4.2.3 Sentence Alignment for MT

Bitext mining has gained significant attention owing to its ability to generate high-quality parallel data for training MT systems as a result of its straightforward approach and wide-ranging utility (NLLB team et al., 2022). In contrast, sentence alignment systems have received limited recognition, despite the potential advantages they offer by capturing not only the broader context found within parallel documents but also the ordering of the sentences therein. As an example, Shi et al. (2021) illustrated the substantial advantages

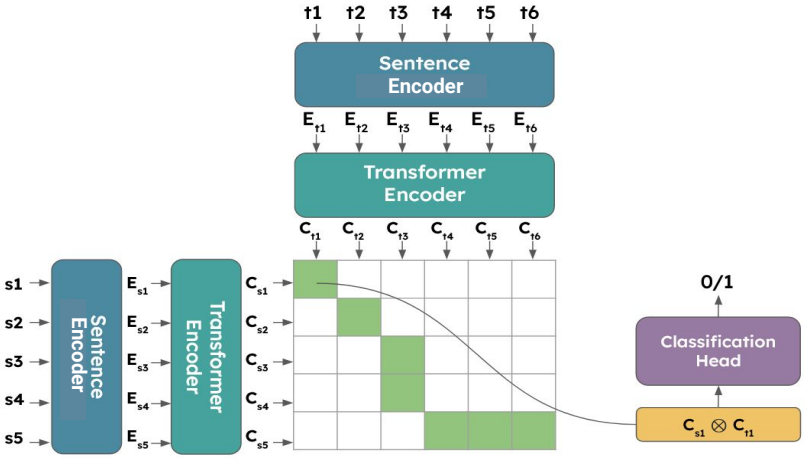

Figure 4.1. The overall architecture of CROCOALIGN. Given a set of source and a target sentences, each individual sentence s_i and t_j is first encoded to obtain sentence embeddings E_{s_i} and E_{t_j}, respectively. Then, the resulting sentence embeddings are given as input to a context encoder, which produces contextualized sentence embeddings, i.e. C_{s_i} and C_{t_j}. Subsequently, the element-wise product of C_{s_i} and C_{t_j} is computed and fed to a linear layer for classification.

that an auxiliary sentence alignment system can yield during the training of MT models. Additional studies have also demonstrated that even sentence alignment systems can be employed effectively to automatically generate data for training MT models (Thompson and Koehn, 2019); however, to the best of our knowledge, we still lack an in-depth investigation of how sentence alignment can be used as a means of producing training data for fine-tuning MT systems on long texts or specific domains/genres.

4.3 CroCoAlign

In this section, we describe CROCOALIGN, a language-agnostic, fully-neural method for aligning sentences between pairs of documents, designed specifically to model context in long texts. Our core intuition is that the embedding of an individual sentence, regardless of how expressive it may be, often lacks crucial information about its surrounding context, such as the preceding and following sentences. The fundamental innovation of CROCOALIGN lies in its ability to capture the document-level continuity of sentence representations.

CROCOALIGN achieves this by first encoding source and target sentences using any sentence encoder (Section 4.3.1). It then enhances the resulting sentence-level representations by employing a novel context encoder, which integrates additional contextual information at the document level (Section 4.3.2). This version maintains flexibility by allowing for various sentence encoders, while keeping the focus on the novelty of context modeling in CROCOALIGN.

Then, it proceeds by determining whether or not a source-target sentence pair is an alignment by feeding the element-wise product of the resulting contextualized sentence embeddings to a multi-label classifier (Section 4.3.3), minimizing the training objective explained in Section 4.3.4. Finally, CROCOALIGN also features a novel two-step procedure, which we refer to as POINTING and RECOVERY, needed at inference time to address the problem of locating the sentences to be aligned and refining the predictions, as described in Section 4.3.5. Figure 4.1 shows the overall architecture.

4.3.1 Cross-lingual Sentence Embeddings

There is a significant body of research that shows that sentence embeddings can be employed effectively in bitext mining to filter and locate parallel sentences across multiple corpora (Schwenk et al., 2021). Given the strong relationship between sentence alignment and bitext mining, we build our approach on pretrained sentence embeddings. Specifically, we exploit the inherent structure of cross-lingual sentence embeddings, where, given two sentences written in different languages but having similar meanings, these are mapped to nearby vectors in the space.

In our experiments, we test two types of sentence embedding approaches. First, we use **parallel static word embeddings**, similar to those employed in Chapter 3. In this approach, each language pair is represented by bilingual word embeddings (Artetxe and Schwenk, 2019). For word alignment, these embeddings allow identifying translation correspondences at the word level by aligning similar words in a shared vector space. Extending this idea to sentences, we generate sentence embeddings by taking the mean of the word embeddings within a sentence, thereby creating a bilingual representation for each sentence. This method leverages the strengths of word alignment techniques while adapting them to capture sentence-level meaning, providing a simple yet effective means of

comparing sentences across languages.

Second, we evaluate the performance of a **language-agnostic cross-lingual sentence transformer**, which allows for a more generalized approach. Unlike bilingual embeddings that require a specific model for each language pair, the sentence transformer can encode sentences across multiple languages into a shared vector space, enabling direct comparisons between sentences without needing pair-specific training. This flexibility makes it suitable for applications involving multiple language pairs without the need for retraining.

We stress that our method is independent of the specific sentence transformer used and supports the integration of both bilingual embeddings and more general multilingual models. Thus, let $s_1, ..., s_n$ and $t_1, ..., t_m$ be a sequence of source and target sentences, respectively. We encode each of these by means of the chosen sentence encoder, generating their respective sentence embeddings $E_{s_1}, ..., E_{s_n}$ and $E_{t_1}, ..., E_{t_m}$. This versatility allows our approach to adapt to different types of cross-lingual sentence embeddings, ensuring robust performance across various sentence alignment scenarios.

4.3.2 Encoding Context Across Sentences

In order to refine $E_{s_1}, ..., E_{s_n}$ and $E_{t_1}, ..., E_{t_m}$, we input these embeddings to a randomly-initialized transformer encoder, which we refer to as context encoder (see Figure 4.1), that, by means of positional embeddings and the attention mechanism, captures the inherent information in the surrounding context. The output of this procedure consists of source and target contextualized sentence embeddings, namely $C_{s_1}, ..., C_{s_n}$ and $C_{t_1}, ..., C_{t_m}$. It is worth noting that it is theoretically possible to obtain a contextual representation of every sentence in a document by encoding all its sentences in a single batch, and that this choice is dictated by hardware constraints.

4.3.3 Classification

Given $C_{s_1}, ..., C_{s_n}$ and $C_{t_1}, ..., C_{t_m}$ from the previous step, we create a matrix $M_{n \times m}$ where the entry M_{ij} contains the element-wise product of the embeddings C_{s_i} and C_{t_j}. The resulting vector is then fed into a classification head in order to output the logits associated with the sentences to be aligned, updating the value of M_{ij}. These values are converted to probabilities by means of the sigmoid activation function and then rounded to binary values

using a threshold of 0.5. This probability is calculated for every pair of source and target sentences in the input batch. As a result, each source sentence can be mapped to zero, one, or multiple target sentences, and vice versa, therefore modeling scenarios such as those in which no sentence in one book has an equivalent in another, or when the alignment is 1-to-many, many-to-1, or many-to-many. Figure 4.1 shows examples of 1-to-1, many-to-1 and 1-to-many alignments, pictured as green cells in the matrix.

4.3.4 Training Objective

The model is trained to maximize the element-wise product between the contextualized embeddings of source and target sentences that correspond to an alignment according to the ground truth. At the same time, the model is also trained to minimize the element-wise product of sentences that should not be aligned. More formally, the loss is computed as follows:

$$\mathcal{L}(M, \hat{M}) = -\frac{1}{n}\frac{1}{m}\sum_{i=1}^{n}\sum_{j=1}^{m}\Big[\hat{M}_{ij} \cdot \log\left(\sigma(M_{ij})\right) + (1 - \hat{M}_{ij}) \cdot \log\left(1 - \sigma(M_{ij})\right)\Big],$$

where M and $\hat{M}$ are both matrices in $\mathbb{R}^{n \times m}$ corresponding to the predicted and gold alignments, respectively, while σ is the sigmoid activation function, applied element-wise to the elements of M.

4.3.5 Identifying Target Contexts and Reducing Noise at Inference Time

While gold alignments are available at training time, allowing us to construct predefined source and target batches to be aligned, this information is missing at inference time. In order to mitigate this issue, we introduce a procedure that we refer to as POINTING, to first identify the source and target batches to be aligned. This procedure uses an approximate nearest neighbor algorithm to find the most closely related target contexts, given a source context. Specifically, using the FAISS algorithm (Johnson et al., 2019), we identify the set of k candidate target sentence embeddings that are closest to a given source embedding E_{s_i}, according to their cosine similarity. Afterwards, in order to identify the correct target embedding, for each candidate we construct a context of N sentences surrounding and including both the source and the pointed target sentences. After obtaining such contexts,

we provide them as input to our model and ask for a prediction. We finally select the target context T with the highest alignment probability, that is, the one with the highest number of alignments.[2]

Additionally, to guarantee alignment between each source sentence and its corresponding target fragments (if any), the subsequent iteration generates a new source context by intersecting $\lceil N/2 \rceil$ sentences with the previous context. This approach makes it possible to align source sentences with target fragments that were originally located outside the boundaries of T.

Due to the strategy just described, there may be some noise in the final prediction. As an example, in different iterations, a source sentence might be aligned with two or more non-adjacent target sentences. In this scenario, we apply a RECOVERY procedure in order to determine which of the non-adjacent target sentences (or combination of target sentences) is most similar to the source sentence. We experiment with different recovery procedures and we explain these in detail in Section 4.4.3.

4.4 Experimental Setup

Our system is developed using the Pytorch Lightning framework[3] and the HuggingFace models library.[4] In order to generate the initial sentence embeddings (Section 4.3.1) for each source and target batch of sentences, we first use **parallel FastText embeddings** (Bojanowski et al., 2017; Smith et al., 2017b) as sentence representations. Specifically, sentences are represented by averaging the FastText embeddings of the words they contain, allowing for a bilingual sentence representation that aligns across the two languages, similar to their application in word alignment tasks.

In addition to the FastText approach, we also employ the **Language-Agnostic BERT Sentence Embeddings** (LaBSE)[5] (Feng et al., 2022) as an alternative method for generating sentence embeddings. Unlike the word-level averaging method, LaBSE provides more

[2]We experiment with $k \in \{5, 10, 20, 100\}$ and observe the best results with $k = 10$. We highlight that we penalize retrieved target contexts that are relatively far away in the document from the source context, regardless of the alignment probability.

[3]`https://www.pytorchlightning.ai/`

[4]`https://huggingface.co/docs/transformers/`

[5]`https://huggingface.co/sentence-transformers/LaBSE`

context-aware representations by encoding entire sentences into a shared multilingual vector space. During training, we keep the weights of the sentence transformer frozen.

For the transformer encoder (Section 4.3.2), we employ a regular 6-layer Transformer initialized with random weights and then trained with six attention heads, six layers and a dropout of 0.2. For the classification head (Section 4.3.3), we use a dropout of 0.2 and the ReLU activation function. We train our system on a RTX 3090 Ti for a maximum of 1.25 million steps and an early stopping mechanism with a patience set to 10. We use the AdamW optimizer with a weight decay of 0.01, a learning rate of 10^{-5}, and a linear scheduler with a warmup of 10% of the maximum number of training steps. We select the best model based on its strict F1 score on the validation set, which demands an exact match between the predicted and the gold alignments. In the following, we describe the dataset we use for our experiments (Section 4.4.1), the baselines we compare with (Section 4.4.2), and the variants of our model (Section 4.4.3).

4.4.1 Dataset

We extract our dataset from the book section of the Opus project website.[6] The website provides a collection of copyright-free books aligned by Andras Farkas.[7] The dataset contains 16 languages, 64 language pairs and a total of 251 parallel books. For each available parallel book, there is a corresponding file specifying the ground truth of the sentences to be aligned according to their IDs. Table 4.1 summarizes dataset information, such as covered languages, number of books written in the source language, number of sentences and tokens. In addition, Table 4.2 shows the details about the occurrences of each alignment type contained in the dataset. For validation and test purposes, we select books whose language pairs appear at least 2 times in the overall corpus. We divide the resulting 20 books into half for validation and half for testing, and we use the remaining 231 books for training purposes.

[6] https://opus.nlpl.eu/Books.php

[7] http://www.farkastranslations.com/

Lang.	# Books	# Sentences	# Tokens
CA	1	5.0K	93.3K
DE	12	71.0K	1.3M
EL	1	1.6K	36.5K
EN	42	0.2M	5.9M
EO	2	2.0K	38.8K
ES	18	0.1M	2.4M
FI	1	3.8K	54.5K
FR	29	0.2M	3.6M
HU	28	0.2M	3.3M
IT	8	36.0K	0.8M
NL	9	55.1K	1.3M
NO	1	4.0K	67.9K
PL	1	3.3K	43.5K
PT	1	1.5K	32.3K
RU	3	27.3K	0.5M
SV	1	3.2K	76.6K
TOTAL	158	0.9M	19.5M

Table 4.1. Statistics of the dataset extracted from the Opus project. # Books, # Sentences, and # Tokens represent the number of books, sentences, and tokens associated with the corresponding language.

Split	Align. Type	# Ann.	%
Train	1-to-1	892,320	77.3
	1-to-0	11,136	1.0
	0-to-1	19,966	1.7
	n-to-1	110,580	9.6
	1-to-m	105,918	9.2
	n-to-m	13,853	1.2
Validation	1-to-1	34,282	75.4
	1-to-0	257	0.6
	0-to-1	669	1.5
	n-to-1	5,955	13.1
	1-to-m	3,597	7.9
	n-to-m	684	1.5
Test	1-to-1	35,162	77.4
	1-to-0	298	0.7
	0-to-1	396	0.9
	n-to-1	6,015	13.2
	1-to-m	2,978	6.6
	n-to-m	601	1.3

Table 4.2. Statistics of the alignment types in our dataset. # Ann. refers to the number of examples annotated with each type of alignment, while % represents the proportion of each alignment type relative to the total.

4.4.2 Baseline Systems

Bleualign. The algorithm proposed by Sennrich and Volk (2011) makes use of an external MT system to guide the alignment based on the BLEU score between the given translation and the target sentence. The alignment can also be cross-validated by entering both source and target translations in order to enhance the performance. The system uses the Gale and Church (1993) algorithm to obtain an initial alignment, and then refines it using MT. However, when the BLEU score between the target sentence and the source translation is not sufficiently high, the algorithm returns the initial alignment. During their experiments, Sennrich and Volk (2011) used an old version of Google Translate as well as a statistical MT system. To ensure a fair comparison, we replace the latter with OPUS-MT, a robust neural MT system developed by Helsinki NLP, available on HuggingFace.[8]

Vecalign. Thompson and Koehn (2019) introduced Vecalign, which is the current state of the art in sentence alignment. We first adapt Vecalign to use **parallel FastText embeddings** (Bojanowski et al., 2017; Smith et al., 2017b), where each sentence is represented as the mean of the FastText word embeddings. This adaptation allows Vecalign to leverage word-level bilingual embeddings for sentence alignment, similar to the method used in word alignment, providing a straightforward approach to generating cross-lingual sentence representations.

In addition to FastText, we also test Vecalign with **LASER bilingual sentence embeddings** (Artetxe and Schwenk, 2019), which provide more sophisticated sentence-level representations directly in a shared vector space. The alignment process in both cases is performed using Fast-Dynamic Time-Warping (see Section 4.2). By comparing the performance of FastText-based and LASER-based embeddings within Vecalign, we assess the effectiveness of different sentence representation techniques for alignment accuracy.

4.4.3 Model Variants

Each version of our model uses the same POINTING strategy, discussed in Section 4.3.5, which employs the LaBSE sentence transformer (Feng et al., 2022). Therefore, in this section, we focus on describing the different variants of our RECOVERY procedure. Let s_i

[8] `https://huggingface.co/Helsinki-NLP`

be a source sentence which has been aligned to non-adjacent target sentences t_j and t_k with $k > j + 1$, constituting an unlikely alignment. We underline that, in general, this procedure can be extended to cases where s_i is aligned with more than two target sentences. For instance, if there are three candidate target sentences, namely t_x, t_y and t_z, with t_x adjacent to t_y, the procedure will be applied to each sentence individually, as well as to groups of adjacent sentences.

CROCOALIGN-FastText. To determine the correct target sentence among the available options, we encode s_i, t_j, and t_k independently using parallel FastText embeddings (Bojanowski et al., 2017; Smith et al., 2017b). These embeddings are obtained by averaging the word embeddings for each sentence, effectively creating a sentence representation. Then, we select the target sentence whose embedding has the highest cosine similarity with the source sentence embedding, leveraging the alignment properties of the parallel FastText space for cross-lingual similarity assessment.

CROCOALIGN-LaBSE. As a means of determining the correct target sentence among the available options, we encode s_i, t_j and t_k independently using LaBSE (Feng et al., 2022). Afterwards, we select the target sentence having the embedding with the highest cosine similarity with the source sentence embedding.

CROCOALIGN-WSD. For the purpose of selecting which target sentence we should keep, we employ a state-of-the-art multilingual Word Sense Disambiguation (WSD) system, namely AMuSE-WSD (Orlando et al., 2022). The system identifies the meanings (i.e. BabelNet synsets) of the words associated with the sentences s_i, t_j, and t_k. Given the synsets associated with each sentence, we select the target sentence with the highest synset intersection. Importantly, we note that this is possible thanks to BabelNet encoding each synset multilingually, i.e. as the set of lexicalizations that are used in different languages to express the given concept (Navigli et al., 2021). Therefore, translated words are ideally assigned the same synset across languages. Our intuition is that, thanks to the assumption that parallel sentences should share the same semantics, our synset intersection approach is likely to select target sentences with the most accurate translation.

CROCOALIGN-LaBSE$_B$. By exploring in detail the output of the RECOVERY procedure explained in CROCOALIGN-LABSE, we observe that the source sentence s_i alone may not provide enough context to make the right choice between t_j and t_k. To better grasp the contextual information, we concatenate s_i together with N surrounding sentences and then encode the resulting text with LaBSE, generating a single sentence embedding. We then do the same for t_j and t_k. Finally, we select the target sentence associated with the embedding having the highest cosine similarity with the source sentence embedding. We experiment with $N \in \{3, 5, 7\}$ and observe the best results on the validation set with $N = 3$. We name this procedure as CROCOALIGN-LABSE$_{Batched}$, or alternatively CROCOALIGN-LABSE$_B$ for short.

4.5 Results

In this section, we present the results obtained by CROCOALIGN and its competitors on the dataset introduced in Section 4.4.1 in terms of strict F1 score.

Quantitative Results. Table 4.3 summarizes the results. Our model, along with its variants, outperforms the baselines 14 times out of 20. In contrast, the best variant, namely CROCOALIGN-LaBSE$_B$, outperforms all the other solutions 11 times out of 20. Indeed, on average, our best model achieves +1.6 F1 points compared to Vecalign, the current state-of-the-art in sentence alignment. However, we also point out that, for specific language pairs, the baselines achieve higher results. For instance, Bleualign reaches the highest score for the EN-ES, EN-NL, and ES-NL language pairs, thanks mainly to the quality of the underlying MT systems for the three languages involved. We note that the absence of results for the Bleualign baseline for some language pairs is attributable to the non-availability of a bilingual MT model from Helsinki NLP for that specific language pair, which is an essential requirement for Bleualign and this, therefore, represents a possible limitation for lower-resource languages. Vecalign, instead, achieves the highest score 3 times out of 20, in the EN-HU, FR-HU, and HU-IT language pairs, respectively, thanks mainly to the strength of its underlying bilingual encoder for the Hungarian language.

The lower performance observed for the FR-IT pair compared to other Romance pairs (e.g., ES-IT or ES-FR) can be primarily attributed to stylistic differences in French

Algorithm	EN-IT	EN-ES	EN-FR	EN-NL	EN-RU	EN-HU	DE-IT	DE-HU	DE-FR	DE-ES	DE-EN
Bleualign	93.7	87.0	87.0	87.8	85.3	92.9	63.7	62.6	67.3	93.2	86.8
Vecalign-FastText	88.6	82.1	83.2	84.5	87.1	88.4	66.1	68.5	71.2	88.9	83.1
Vecalign-LASER	95.4	85.7	87.1	87.6	91.3	**96.3**	70.1	76.3	70.7	94.4	88.4
Vecalign-LaBSE	95.7	**89.1**	88.4	**90.6**	92.0	95.7	71.3	72.7	70.6	94.7	87.1
CROCOALIGN-FastText	90.1	83.8	85.7	86.9	89.5	89.9	67.8	69.7	72.4	90.4	84.5
CROCOALIGN-WSD	96.3	77.5	95.7	82.8	**95.7**	90.4	75.9	78.9	68.9	97.8	94.4
CROCOALIGN-LaBSE	96.0	76.8	95.5	82.0	95.4	89.8	75.2	78.8	**78.7**	97.5	94.2
CROCOALIGN-LaBSE$_B$	**96.4**	77.7	**97.0**	83.8	**95.7**	90.8	**80.0**	**81.6**	75.8	**98.9**	**95.6**

(a)

Algorithm	ES-IT	ES-FR	ES-NL	ES-HU	FR-IT	FR-NL	FR-HU	HU-IT	HU-NL	Macro Average
Bleualign	72.8	88.8	86.9	—	—	—	74.4	—	—	82.0
Vecalign-FastText	80.1	84.2	83.1	91.2	61.4	77.8	82.1	83.2	81.5	82.2
Vecalign-LASER	86.4	91.2	86.5	97.8	65.9	82.7	**88.1**	89.9	86.9	85.9
Vecalign-LaBSE	85.7	88.0	**87.8**	92.7	65.6	82.3	85.0	**90.3**	85.1	85.5
CROCOALIGN-FastText	81.4	85.6	84.3	92.8	62.9	79.5	83.5	84.5	82.9	83.7
CROCOALIGN-WSD	87.3	**93.9**	85.2	97.5	**66.6**	90.7	72.9	85.9	88.1	86.1
CROCOALIGN-LaBSE	87.1	93.6	85.0	97.5	65.8	90.4	75.5	85.2	88.0	85.7
CROCOALIGN-LaBSE$_B$	**89.5**	92.7	86.0	**98.9**	66.2	**92.3**	76.1	85.1	**88.9**	**87.5**

(b)

Table 4.3 (a) and (b) present the strict F1 scores (%) of CROCOALIGN and its variants (bottom) compared with baseline systems (top) across language pairs. In (b), the final column shows the average F1 across all pairs. **Bold** indicates the best results, and underline the second-best.

translations, especially in literary texts. French translations often favor more elaborate and nuanced phrasing, while Italian tends to maintain a more direct structure. For example, a simple Italian sentence like *"Tre giorni fa"* (literal: *"Three days ago"*) might be translated in French as *"Il y a trois jours"* (literal: *"It there has three days"*), reflecting a stylistic choice rather than a syntactic necessity. These stylistic divergences introduce additional complexity for alignment models relying on sentence-level embeddings.

We also include an evaluation of CROCOALIGN and Vecalign using parallel FastText embeddings. Unlike contextualized models like LaBSE or LASER, FastText embeddings represent a sentence as the mean of its word embeddings. While this approach provides a straightforward and fast way to obtain sentence-level representations, it cannot account for word order and nuanced contextual information within a sentence. As a result, FastText-based alignments tend to be less effective at distinguishing between sentences with similar words but different meanings. Consequently, the performance of CROCOALIGN-FastText and Vecalign-FastText is generally lower than that of models leveraging contextualized embeddings. Specifically, these models struggle more with complex alignment scenarios where the precise context of each word matters, as shown by their lower average F1 scores across various language pairs.

Finally, even though Vecalign employs LASER as the underlying sentence encoder, to ensure a fair comparison, we also evaluate it using LaBSE. The main difference between the two encoders is that the former is designed as a bilingual model—requiring a distinct model for each language pair—while the latter is language agnostic. As shown in Table 4.3, we observe that replacing LASER with LaBSE has no beneficial impact on Vecalign's performance. On the contrary, the results of Vecalign with LaBSE are lower than those with LASER, i.e., 85.5 versus 85.9 in F1 score on average across all languages.

Alignment Analysis. In addition to the quantitative results presented in the previous paragraph, we performed an analysis of the accuracy of different alignment types, comparing our best model with Vecalign. In Figure 4.2 we report the accuracy, expressed as a percentage, of the number of times a specific type of alignment is predicted correctly by the two systems. From the results we can see that, on average, both CROCOALIGN and Vecalign perform similarly when tested on 1-to-1 up to 1-to-7 alignments. However, when presented with

particularly challenging types of alignment (upper entries in Figure 4.2), which can happen frequently in long texts, our system consistently outperforms its competitor. Moreover, CROCOALIGN is also more resilient to other very common situations in long texts where, given a sentence, there is no equivalent in its corresponding parallel document (1-to-0 and 0-to-1 alignments in Figure 4.2).

Inference Speed. When assessing sentence alignment systems, it is crucial to consider their inference speed, especially given their typical application to extensive datasets. In this context, we conducted a comparative analysis between CROCOALIGN and Vecalign, our primary competitor. Notably, Vecalign employs a combination of dynamic programming and sentence embeddings for a fast alignment process. However, it necessitates a preprocessing step where sentences from source and target documents undergo a complex encoding process using a sentence transformer. Indeed, the algorithm requires the encoding of clusters of adjacent sentences to identify many-to-many alignments, introducing an overhead in the overall process before the execution of the alignment step.

In contrast, using FastText embeddings with CROCOALIGN offers a substantial speed advantage. Unlike contextualized models like LASER or LaBSE, FastText embeddings rely on precomputed word vectors that are stored in a lookup table. This allows for rapid generation of sentence embeddings, as it only involves averaging the vectors of words in a sentence, without the need for GPU processing. This makes CROCOALIGN-FastText significantly faster than its counterparts using context-sensitive models, making it especially suitable for scenarios where computational resources are limited or where rapid alignment is a priority.

However, it is important to note that using FastText embeddings requires having pre-aligned embeddings for each specific language pair. This can limit flexibility when dealing with low-resource languages or new language pairs where precomputed bilingual embeddings may not be readily available. In such cases, training or obtaining the required word embeddings can add an extra step to the overall setup process.

As a consequence, CROCOALIGN is $2.7\times$ faster than Vecalign-LASER and $3.3\times$ faster than Vecalign-LaBSE. On a single GTX 1080 Ti, CROCOALIGN requires 84 seconds on average (9.49 of which are devoted to the generation of sentence embeddings and 75.12

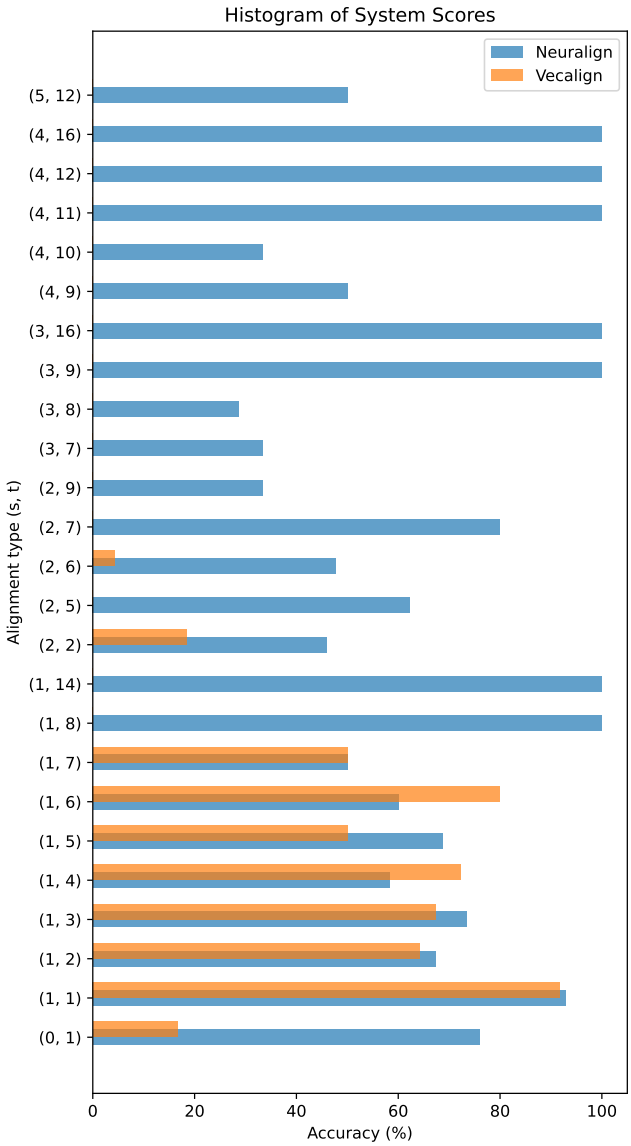

Figure 4.2. Histogram showing the accuracy (%) on one-to-many, many-to-one, and many-to-many alignments. Alignments on the Y-axis are clustered, e.g., `(1,2)` includes 1-to-2 and 2-to-1 alignments.

to produce the final alignment) to align a pair of parallel books in our dataset. In contrast, Vecalign requires 224 seconds on average to encode both source and target documents (112 seconds for a single document) and 3 seconds to find the optimal alignment, for a total of 227 seconds. As a rough estimate, when a single GPU is adopted, CROCOALIGN can align around 1000 pairs of books and 50 million tokens in a day, compared to the 380 pairs of books and 18.5 million tokens of Vecalign.

This version balances the discussion of speed advantages with the practical consideration of needing precomputed FastText embeddings for each language pair.

4.6 Experiments on Machine Translation

Sentence alignment serves as a fundamental tool in MT systems, as it can be used for the creation of parallel datasets for training purposes. Here, we demonstrate the effectiveness of CROCOALIGN for the automatic creation of a high-quality parallel fine-tuning set that can be used to adapt an existing MT system to specific domains. Our hypothesis is that, with a high-quality alignment, we only require a small amount of parallel sentences to significantly boost the performance of a pretrained MT model.

Experimental setup We compare the performance of a strong MT baseline when used out of the box with the performance of the same system when fine-tuned on the parallel data created with CROCOALIGN. We evaluate the impact of our data on the bilingual MT models from the OPUS-MT family, for a total of 15 bilingual models. We fine-tune each MT model on the alignments produced by CROCOALIGN-LaBSE$_B$ when applied on the validation set of the dataset introduced in Section 4.4.1. Finally, we report the sacreBLEU scores obtained by the MT models on the corresponding test sets, before and after fine-tuning.

Results Table 4.4 provides an overview of the results obtained by the OPUS-MT models when used to translate books with and without fine-tuning. We can observe a significant increase in terms of sacreBLEU across all 15 language pairs, resulting in an average improvement of 4.1 points with a minimum improvement of 2.0 points in DE-HU and DE-IT and a maximum improvement of 7.5 points in EN-FR. These results further demonstrate the high quality of our automatically-aligned data, which leads autoregressive models to better

Language Pair	Fine-Tuning	BLEU Score	Δ
DE-ES	✗	23.9	+4.4
	✓	**28.3**	
DE-EN	✗	20.7	+2.1
	✓	**22.8**	
DE-FR	✗	9.2	+4.1
	✓	**13.3**	
DE-HU	✗	8.5	+2.0
	✓	**10.5**	
DE-IT	✗	4.2	+2.0
	✓	**6.2**	
EN-ES	✗	17.8	+6.1
	✓	**23.9**	
EN-FR	✗	33.6	+7.5
	✓	**41.1**	
EN-HU	✗	11.8	+3.0
	✓	**14.8**	
EN-IT	✗	17.9	+6.0
	✓	**23.9**	
EN-NL	✗	15.6	+6.9
	✓	**22.5**	
EN-RU	✗	20.6	+4.9
	✓	**25.5**	
ES-FR	✗	19.5	+2.6
	✓	**22.1**	
ES-IT	✗	12.7	+2.1
	✓	**14.8**	
ES-NL	✗	10.4	+5.2
	✓	**15.6**	
FR-HU	✗	8.2	+2.8
	✓	**11.0**	

Table 4.4. sacreBLEU results on the machine translation downstream task. Each bilingual model is evaluated with and without fine-tuning over the test split of our dataset. The fine-tuning data is produced by applying CROCOALIGN on the validation split.

translate domain-specific texts.

4.7 Conclusion

In this chapter, we presented CROCOALIGN, the first fully-neural and language-agnostic architecture designed to perform sentence alignment in very long texts. Building upon the concepts explored in word-level alignment, we shifted our focus to the challenges and opportunities of sentence-level representations. The strength of our approach lies in its ability to create enhanced sentence representations by leveraging their surrounding context, using a novel encoder that captures cross-sentence information, including the position and meaning of a sentence relative to those preceding and following it.

Our experiments on sentence alignment in books—which involve extremely long contexts and frequent many-to-many alignments—demonstrated that CROCOALIGN outperforms the previous state-of-the-art method, Vecalign, across 20 language pairs, achieving an average improvement of +1.6 points in F1 score. Furthermore, we explored the practical

impact of our method by showing that parallel data produced by CROCOALIGN can be used to fine-tune machine translation (MT) systems, leading to significant improvements in domain-specific translations, with an average gain of +4.1 points in BLEU score.

In addition to the performance gains, we highlighted the flexibility of CROCOALIGN to use different types of sentence embeddings, such as LaBSE and even simpler methods like parallel FastText embeddings, to adapt to various data and resource scenarios. This flexibility, combined with the ability to align sentences in a language-agnostic manner, makes CROCOALIGN particularly suited for creating high-quality parallel data for low-resource language pairs.

By progressing from word-level to sentence-level alignment, this chapter illustrates the evolving nature of representations and the benefits of contextual understanding. As we move forward, the next chapter will explore how these sentence-level insights can be combined with the detailed knowledge obtained from word-level representations. Specifically, we will focus on the complex task of sense disambiguation, aiming to unify word and sentence-level information to further advance our understanding of semantics and enrich NLP systems' capacity to understand nuanced meanings.

∼ This page was intentionally left blank ∼

Chapter 5

Combining Word and Sentence Representations for Enhanced Semantic Understanding

Abstract

Building on our exploration of word and sentence representations, this chapter presents an integrated approach that combines both types for deeper semantic understanding. Word embeddings capture specific lexical nuances, while sentence embeddings provide broader context. However, many tasks—such as document retrieval, entity linking—can leverage the advantages of a hybrid approach that balances both local details and global context.

We address this need through the retriever-reader paradigm, where sentence embeddings efficiently retrieve relevant sections, and word embeddings offer precise term-level analysis. We introduce a novel architecture for Word Sense Linking (WSL), which adapts traditional disambiguation methods to real-world applications without predefined spans or sense mappings.

Our experiments show that combining word and sentence representations substantially improves robustness and precision, surpassing standalone approaches. This chapter sets the foundation for the next step: exploring synthetic data generation with Large Language Models to refine these embeddings further.

5.1 Introduction

In previous chapters, we examined how word and sentence representations can be applied to various NLP tasks, including word alignment and sentence alignment. These representations offer unique strengths: word embeddings capture fine-grained semantic nuances, while sentence embeddings provide a broader contextual understanding. However, many real-world tasks require a deeper integration of these two levels of representation to capture both the local details and the global context.

Integrating word and sentence embeddings can significantly enhance the understanding of complex language phenomena. One particularly effective approach to combining these representations is the retriever-reader paradigm (Chen et al., 2017). This architecture is especially useful in scenarios requiring broad contextual understanding and fine-grained semantic distinctions. Relying solely on sentence or word embeddings has inherent limitations:

- **Sentence representations:** Capture overall context and relationships between sentences, making them efficient for retrieving relevant passages from large collections (Karpukhin et al., 2020). However, they may miss specific word-level nuances, such as differentiating between polysemous terms like "bank" (financial institution vs. riverbank).

- **Word representations:** Capture subtle distinctions between individual terms, useful for precise tasks like disambiguation. However, when dealing with larger contexts, word representations can have their semantic meaning diluted, making it harder to retain relevant contextual information, thus limiting their effectiveness in understanding the broader meaning of a passage. (Liu et al., 2024)

By combining both approaches in the retriever-reader paradigm, we can leverage the strengths of each: **sentence representations** help quickly identify relevant portions of text, while **word representations** enable detailed, accurate analysis of specific terms.

The retriever-reader architecture has demonstrated significant promise across various NLP tasks, especially those requiring a balance between broad contextual understanding and fine-grained semantic analysis. For instance, it is widely used in Open-Domain Question Answering (QA) (Chen et al., 2017; Karpukhin et al., 2020) and Document Retrieval

for Fact-Checking (Thorne et al., 2018). In these tasks, the retriever uses sentence-level representations to identify contextually relevant sections from large datasets, while the reader model refines this information by focusing on word-level details for precise answers or evidence.

Our focus lies particularly on semantics-related tasks, such as **Entity Linking** (Rao et al., 2013). Entity Linking aims to match mentions of named entities (e.g., people, places, organizations) in text to their corresponding entries in a knowledge base. In this process, the retriever stage first identifies potential entity candidates from a vast knowledge base using sentence-level similarities to filter relevant entities. The reader stage then refines the selection, focusing on word-level distinctions, such as the specific context of the mention, to select the most accurate entity based on detailed matching (Karpukhin et al., 2020; Orlando et al., 2024).

While named entities are fundamental to many downstream applications, they represent only a subset of the semantic units in natural language texts. Many real-world scenarios require the disambiguation of broader concepts beyond named entities. This is where **Word Sense Disambiguation (WSD)** comes into play. WSD is tasked with determining the correct meaning of a word based on its context. (Navigli, 2009, 2012)

Leveraging the advances in pretrained transformer architectures (Devlin et al., 2019), WSD systems have nowadays reached performances on several evaluation benchmarks that are on par with their estimated inter-annotator agreement (Bevilacqua and Navigli, 2020; Barba et al., 2021c). However, despite these advances, the task is still well known for struggling to find downstream applications. We argue that one of the possible causes is the difficulty of applying WSD in unconstrained settings due to its heavy assumptions. Indeed, three requirements need to be met for a generic state-of-the-art system to perform WSD on some input text:

R1) a sense inventory, that is, a semantic resource providing a comprehensive list of all the senses of interest, must be provided,

R2) the list of spans to disambiguate in the input text must have already been identified, and

R3) an oracle that pairs each span to its set of possible senses, realized through a manually

curated word-to-sense mapping, must be available.

While the first condition is intrinsic to the disambiguation objective and, thus, unavoidable, we argue that the second two (i.e. R2 and R3) are system-specific assumptions that can be relaxed. We formulate this idea into a new task called Word Sense Linking (WSL), which more accurately reflects the conditions of downstream applications such as Neural Machine Translation (Liu et al., 2018; Iyer et al., 2023), Information Extraction (Moro and Navigli, 2013; Delli Bovi et al., 2015) and the enrichment of contextual representations in neural models (Peters et al., 2019). In WSL, given an input text and a reference sense inventory, systems must identify which spans to disambiguate and link them to their most suitable meaning in the sense inventory.

Similarly to Entity Linking (Broscheit, 2019; Orlando et al., 2024), WSL can be split into three simpler subtasks, with traditional WSD taking place after two initial stages of Concept Detection (CD), that is, the identification of the spans to be disambiguated in the input text (R2), and Candidate Generation (CG), the generation of a list of sense candidates for each target span (R3). For example, given some reference sense inventory and the sentence *"Bus drivers drive buses for a living."*, Concept Detection might identify `[bus drivers, drive, buses, living]` as the spans to disambiguate, while Candidate Generation might provide `[vehicle, electrical conductor]` as the sense candidates for *buses*.

In this chapter, we first formally introduce the task of Word Sense Linking, along with its three components, and, then, put forward a novel architecture for the task, based on the retriever-reader paradigm (Karpukhin et al., 2020). We thoroughly study the behavior of our architecture and that of state-of-the-art WSD systems, once they have been scaled to WSL, as we iteratively relax the sandboxed assumptions of WSD, starting with R2 and continuing with R3. Our analysis highlights a number of important – yet neglected – challenges when it comes to performing disambiguation in unconstrained settings. In particular, straightforward and natural extensions of WSD systems to WSL result in large performance drops. In contrast, our model demonstrates considerably more robustness and consistently outperforms such extensions of WSD systems by a large margin. The contributions of this chapter are therefore as follows:

- We introduce the task of (Bejgu et al., 2024, Word Sense Linking), which we believe

to better represent the settings of downstream applications for WSD.

- We introduce a Word Sense Linking evaluation dataset, enriching the *de-facto* standard WSD benchmark (Raganato et al., 2017).
- We put forward a novel flexible architecture for this task and evaluate, in multiple settings, both its behavior and that of state-of-the-art WSD systems scaled to WSL.
- Overall, our findings underline several crucial yet neglected challenges when scaling WSD systems to a real-world scenario.

5.2 Word Sense Linking

Word Sense Linking is identifying and disambiguating all the spans of an input text with their most suitable senses chosen from a reference inventory. Formally, let t be the input text, with $t_1, \ldots, t_{|t|}$ being its words, and I the reference inventory, containing a set of senses. Then, a WSL system can be represented as a function f that takes as input the tuple (t, I) and outputs a list of triples $[(s_1, e_1, g_1), \ldots, (s_n, e_n, g_n)]$ where each triple (s_i, e_i, g_i), $i \in [1, n]$, represents a disambiguated span, with s_i and e_i being the start and end token index of the span, and $g_i \in I$ representing the corresponding sense chosen from the inventory. Conceptually, WSL can be divided into three subtasks, namely, Concept Detection, Candidate Generation, and Word Sense Disambiguation.

Concept Detection (CD) is the task of identifying the spans to disambiguate in an input text t given a reference inventory I. A Concept Detection system can be represented as a function that takes as input the tuple (t, I) and outputs a list of pairs $[(s_1, e_1), \ldots, (s_n, e_n)]$, each marking the boundaries of a span to disambiguate. Specifically, $\forall i \in [1, n]$, s_i and e_i are the start and end token index of the i-th span to disambiguate.

Candidate Generation (CG) is the task of generating a set of possible candidate senses that an input span occurring in some text t can assume given the reference inventory I. A Candidate Generation system can be represented as a function that takes as input t, I and the start and end token indices (s, e) of the span, and outputs a set of sense candidates $PC \subseteq I$, that the span can assume.

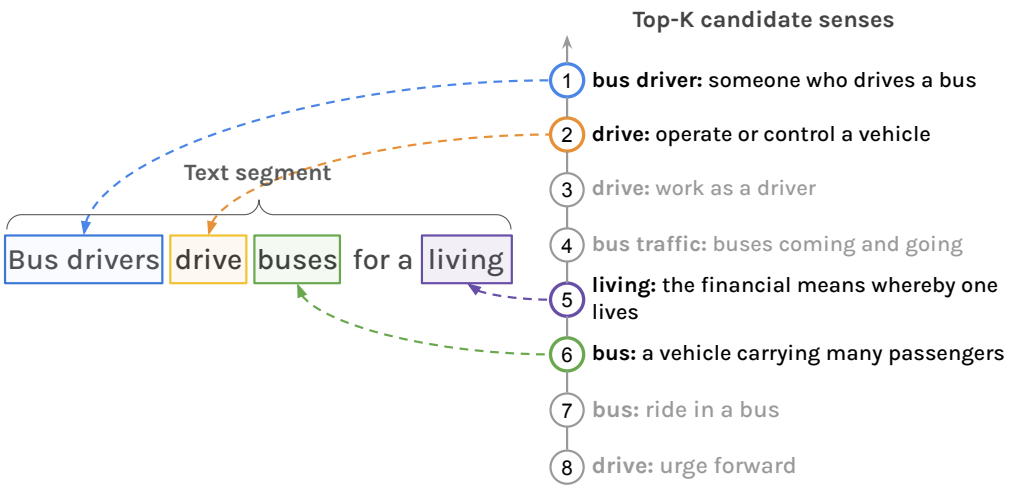

Figure 5.1. Our WSL process. First, the retriever identifies the top-k candidate senses (Candidate Generation). Then, the reader identifies the spans to be disambiguated (Concept Detection) and pairs each of these with their most suitable sense (Word Sense Disambiguation).

Word Sense Disambiguation (WSD) aims at identifying, for each target span occurring in an input text t, the most suitable sense among a set of possible candidates. A WSD system can be represented as a function that takes as input t, I, a list of span indices $[(s_1, e_1), \ldots, (s_n, e_n)]$ and corresponding sets of possible candidates $[PC_1, \ldots, PC_n]$. It then outputs a list of triples $[(s_1, e_1, g_1), \ldots, (s_n, e_n, g_n)]$ where, $\forall i \in [1, n]$, each span (s_i, e_i) is paired with the sense g_i chosen by the system among the candidates provided in PC_i.

Finally it is easy to see that a WSL system can be built by concatenating, in cascade, a Concept Detection, a Candidate Generation, and a Word Sense Disambiguation module. This structure not only facilitates the extension of WSD systems to WSL but also serves as a flexible framework rather than a strict recipe. In fact, as we will show in the next section, the architecture we put forward does not follow this flow and inverts the CD and CG steps.

5.3 Model

The flow *CD-then-CG* presents an intrinsic limitation: each span identified by Concept Detection ought to correspond to the occurrence of a specific sense candidate among those produced by Candidate Generation and, yet, CG occurs after CD. That is, we need to identify

the span we will, later on, link to a specific sense without actually knowing this sense. To overcome this limitation, inspired by the work presented by Zhang et al. (2022b) for Entity Linking, we invert the CD and CG steps, and propose a novel flexible architecture for WSL based on the retriever-reader paradigm. In this section, we first outline its formulation (Section 5.3.1) and, then, discuss its two components, namely the retriever (Section 5.3.2) and the reader (Section 5.3.3).

5.3.1 Formulation

Starting from the input text t and the reference inventory I, our formulation is as follows. First, we perform CG on the entire input text, producing an ordered list of unique candidates $PC(t) \subseteq I$, each coming from I and likely to represent the meaning of some arbitrary span in t. Then, moving to CD and using vector representations contextualized on both t and $PC(t)$, we let a classifier identify the start and end token indices of every span in t; this operation results in a list of n tuples $[(s_1, e_1), \ldots, (s_n, e_n)]$. Finally, we perform WSD on the identified spans, pairing each (s, e) with its most suitable sense $g^* \in PC(t)$.[1] Figure 5.1 reports a visual outline of this process.

We implement our formulation with transformer-based architectures that operate on textual inputs. However, this makes it necessary for our inputs to have such representations and, while t inherently satisfies this requirement, the same does not hold for the senses in I. To overcome this issue, for each sense $g \in I$, we define a textual representation, which we build by concatenating its lemmas[2], provided through the mapping from word to possible senses, and its textual definition in I.[3]

Finally, we note that t can be arbitrarily long. Consequently, providing the entire t as input to our formulation can be challenging, both computationally and from a modeling perspective. To overcome this, we adopt a sliding window approach, with window size w and stride τ, and actually feed, to the previous steps, single chunks of t, rather than the entire sequence. Once all chunks have been processed for t, we retain only *conflict-free predictions*, i.e., every triple (s, e, g^*) such that the processing of every chunk that includes

[1] We extend the $\in$ operator to work on lists as well.

[2] If a mapping between senses and words is available.

[3] Although we did not explicitly mention it before, it is customary for sense inventories to provide, for each of their senses, a textual definition (*gloss*) that defines its meaning.

(s, e) in its window always results in assigning the same g^* to (s, e). Henceforth, with no loss of generality, we replace our input, and the meaning of notation t, with a generic chunk, rather than the entire sequence.

5.3.2 Retriever

We implement the initial Candidate Generation step via dense passage retrieval (Karpukhin et al., 2020), using a transformer-based retriever consisting of an encoder E to produce a dense representation of text passages and senses. Starting from the input text t and the inventory I, we use E to compute a vector representation v_t for t, and v_g for every sense $g \in I$. Then, we use the dot product $v_t \cdot v_g$ to rank all the senses in I and, finally, extract the top k among these. The resulting $g^1, \ldots, g^k$ senses constitute our sense candidate set $PC(t)$ for t.

To train our model, we use a multi-label variant of noise contrastive estimation (Zhang et al., 2022b, NCE), maximizing the following objective:

$$\sum_{g \in \Gamma(t)} \log \frac{\exp(v_t \cdot v_g)}{\exp(v_t \cdot v_g) + \sum_{g' \in \mathrm{N(t)}} \exp(v_t \cdot v_{g'})}$$

where $\Gamma(t) \subset I$ is the set of gold senses appearing in t and $N(t) \subset I \setminus \Gamma(t)$ represents a selection of negative examples. Considering the crucial role of $N(t)$ in retriever-based architectures (Karpukhin et al., 2020), we adopt a strategy aimed at selecting adequately informative negative examples, while, at the same time, accounting for the presence of unannotated tokens. Starting from $I \setminus \Gamma(t)$, we first discard the senses that never appear in the training set or whose part of speech does not match any sense-annotated span in t. Then, denoting by $I^R(t)$ the set of resulting senses in I for t, we define $N(t)$ as the union of the two sets $N_1(t)$, a collection of hard negatives (Gillick et al., 2019) in $I^R(t)$, and $N_2(t)$, a subsample of $I^R(t)$ that aims at counterbalancing the bias towards the most frequent senses in the training corpus. Specifically, we build $N_1(t)$ by selecting the ν_1 senses in $I^R(t)$ to which the retriever assigns the highest score, and $N_2(t)$ constructed using ν_2 gold senses from other samples in the same mini-batch.

5.3.3 Reader

Having identified the sense candidates $PC(t) = g^1, \ldots, g^k$, we now describe the remaining Concept Detection and WSD steps, which we formulate as a multi-task multi-label classification problem. To this end, we concatenate t and $g^1, \ldots, g^k$ into a single sequence m, prepending a special symbol $[S_i]$ for each sense candidate g^i:

$$m = t\ [S_1]\ g^1\ \ldots\ [S_k]\ g^k \tag{5.1}$$

We feed m to a transformer encoder, producing a series of vector representations for the tokens in m; let $h_1, \ldots, h_{|t|}$ be the representations corresponding to the tokens in t, and h_g the one associated with the special symbol of a generic sense candidate $g \in PC(t)$. With these contextualized representations, we realize CD and WSD as follows.

We begin with CD, identifying the spans to disambiguate. Specifically, we apply two classification heads, namely H_{start} and H_{end}, on each representation to determine whether the corresponding token is a start or end of some span in t; both heads consist of two linear transformations with a ReLU activation in between. However, their behavior – and input – differ: while H_{start} receives single token representations, H_{end} operates at span level. That is, once H_{start} has identified some index s as a span start, to determine whether some other index $e \geq s$ is its end, H_{end} takes as input $[h_s; h_e]$, that is, the concatenation of h_s and h_e. Once completed, this step results in a list of span indices $[(s_1, e_1), \ldots, (s_n, e_n)]$.

Then, moving to WSD, we let each extracted span (s, e) identify the sense $g^* \in PC(t)$ it refers to. We start by computing the following vectors:

$$h'_{se} = [f_1(h_s); f_2(h_e)]$$
$$h'_g = [f_1(h_g); f_2(h_g)] \ \ \forall g \in PC(t)$$

where f_1 and f_2 are both functions consisting of two identical but independent linear transformations, interleaved by a ReLU activation. Then, we pair the span (s, e) with the sense $g^* = \text{argmax}_{g \in PC(t)}\ h'_{se} \cdot h'_g$. We replicate this strategy for all the extracted spans, hence eventually producing as output $[(s_1, e_1, g_1^*), \ldots, (s_n, e_n, g_n^*)]$. Our reader is trained by jointly maximizing three cross-entropy objectives, respectively over

i) the gold start indices in t,

ii) the gold end indices in t and

iii) the gold sense for every span in t.

We note that our architecture presents a number of interesting properties. First, since $h_1, \ldots, h_{|t|}$ are vectors contextualized over the entire input sequence m (Equation 5.1), which includes $g^1, \ldots, g^k$, the choice of the extracted spans does indeed take into account the sense candidates available. Second, this contextualization is not limited to $h_1, \ldots, h_{|t|}$ but, in fact, applies to $h_{g^1}, \ldots, h_{g^k}$ as well. This means that the sense candidates are also contextualized on each other, allowing for better representations, as in Barba et al. (2021a). Third, computationally speaking, the only demanding operation corresponds to the encoding of $h_1, \ldots, h_{|t|}$. However, since this operation depends only on t and $PC(t)$, we can disambiguate all the spans in t in a single forward pass, which makes it very fast and hence suited for downstream applications.

5.4 Evaluation

We now investigate the effectiveness of our model and how it relates to the R2 and R3 assumptions of WSD (Section 5.1). To this end, we first outline its implementation details (Section 5.4.1) and the novel dataset introduced for the WSD and WSL evaluation (Section 5.4.2), which will remain unchanged throughout our experiments. Then, we evaluate it on plain English WSD (Section 5.5.1), so as to set an initial reference for what regards its disambiguation capabilities. Finally, we move to WSL and gradually relax these sandboxed assumptions, dropping R2 (Section 5.5.2) and examining different relaxations on R3 (Section 5.5.3).

5.4.1 Model Details

Formulation Hyperparameters We use $w = 16$ and $\tau = 8$ for our sliding window approach. This ensures that each window captures sufficient context for an effective disambiguation while maintaining a manageable overlap between consecutive windows.

Retriever We employ mean pooling over $E5_{base}$ (Wang et al., 2022), a pretrained transformer-based architecture, for our retriever encoder E. The entire system is trained

with a batch size of 16 input texts for 150 000 steps using AdamW (Loshchilov and Hutter, 2019), with 20% warm-up and $2 \cdot 10^{-5}$ learning rate, and setting $k = 100$. At training time, we construct the sense candidate set by including the gold senses, $\nu_1 = 5\%$ of negatives from $N_1(t)$ and the remaining ν_2 from $N_2(t)$.

Reader We use the base version of DeBERTa-v3 (He et al., 2021a) as the transformer-based encoder in our reader. The four linear transformations, namely two for span detection and two for Word Sense Disambiguation, present the same intermediate dimensionality, that is, 768; instead, the final dimension is 1 for the span detection mappings, as they are classification heads, but 768 for the WSD mappings. The overall system is trained with a batch size of 4096 tokens for 100 000 steps using AdamW (Loshchilov and Hutter, 2019) with a learning rate of 10^{-4} and a layerwise decay rate of 0.8.

5.4.2 WSL Benchmark

The framework presented by Raganato et al. (2017) represents the de facto standard benchmark for WSD and is based on the WordNet sense inventory (Miller et al., 1990). Specifically, it is composed of Senseval-2 (Edmonds and Cotton, 2001, **SE02**), Senseval-3 (Snyder and Palmer, 2004, **SE03**), SemEval-2007 (Pradhan et al., 2007, **SE07**), SemEval-2013 (Navigli et al., 2013, **SE13**) and SemEval-2015 (Moro and Navigli, 2015, **SE15**). The original task datasets might include spans of text that contain content words but were not assigned any specific meaning by annotators from among those contained in the reference sense inventory. This could have occurred for a variety of reasons, such as in the case of SE13, where only nouns were chosen to be annotated. Whereas for the WSD setting the absence of this information does not pose a problem for evaluation, for WSL it renders evaluation impossible. Specifically, we cannot in such case measure the precision of a WSL system as we cannot discern between wrongly predicted spans and annotation omissions. To address this issue and overcome the lack of a WSL-specific dataset, we introduce a dedicated evaluation resource aimed at bridging the annotation gap, not just in terms of precision, but also in terms of recall. We comprehensively annotated the standard WSD evaluation datasets, increasing the annotation count from 7253 to 11623 annotations, and resulting in the complete coverage of all the content words.

This substantial increase in annotations allows for a comprehensive benchmark, facilitating future research in the field by providing a more robust framework for evaluating the precision, recall, and F1 of WSL systems.

Annotation process We have annotated the standard WSD evaluation dataset (i.e., ALL, the one utilized in the previously presented WSD evaluation framework (Raganato et al., 2017)). This process aimed to ensure a comprehensive and accurate representation of terms and their meanings, using WordNet as the sense inventory. We selected these guidelines for the annotators:

1. **Multiwords:** Annotators marked terms such as "lung cancer" as single entities, focusing on those recognized in WordNet with contextually coherent meanings.

2. **Sub-words:** Annotators also marked the individual components of multiword expressions, but only when the sub-words had coherent meanings within the sentence context. This dual-level annotation strategy captured both the general and specific meanings of terms.

3. **Non-content Words**: Annotators excluded non-content words, such as auxiliary verbs, from annotation. These words are essential for grammatical structure but do not carry any semantic weight.

Employing WordNet as the sense inventory facilitated a uniform and precise approach to annotation across the entire dataset and maintained consistency with WSD tasks, ensuring alignment with established standards and methodologies in the field.

Due to the complexity of the task, a single expert linguist, who has a robust background in the annotation of lexical-semantic tasks, conducted the majority of the annotation work. Nonetheless, to validate the reliability and consistency of the annotation process, we conducted an inter-annotator agreement evaluation. This involved a subset of the dataset, constituted by 20 sentences independently annotated by three different expert linguists,[4] resulting in 222 distinct annotations. The agreement among annotators was measured using Cohen's kappa statistic, which yielded a score of 0.83, interpreted as an almost perfect level of agreement, showing a high degree of annotation consistency. These results underscore

[4]We paid the annotators according to the standard salary for their geographical location.

	Dataset	Sentences	Tokens	Instances	New Instances
Train	SemCor	37176	820410	226036	-
	SemCor$_C$	37176	820410	359763	-
Evaluation	semeval2007	135	3219	455	941 (+206%)
	semeval2013	306	8533	1644	2194 (+133%)
	semeval2015	138	2643	1022	157 (+15%)
	senseval2	242	5829	2282	444 (+19%)
	senseval3	352	5640	1850	634 (+34%)
	all	1173	25864	7253	4370 (+60%)

Table 5.1. Statistics for training and evaluation corpora. The columns represent the number of sentences, the total number of tokens, the number of annotated terms, and the number of newly annotated instances added in each dataset.

the robustness and reliability of our annotation methodology, despite the inherent challenges of subjective interpretation and the detailed demands of manual annotation.

5.5 Dataset Details

We introduced a substantial addition of new instances across various datasets, achieving an overall 60% increase as shown in Figure 5.3.

Before our annotation process, significant gaps were present in the POS tags across various datasets, with certain categories, such as verbs, adjectives, and adverbs in the semeval2013 dataset, and adjectives and adverbs in the semeval2007 dataset, being completely absent. This is evident from the missing data points in Figure 5.3. Filling these gaps is crucial for constructing robust and comprehensive evaluation benchmarks. Incomplete datasets can lead to evaluations that fail to measure the true capabilities of language processing systems in real-world scenarios. Our annotation efforts were, therefore, critical in ensuring that all POS categories were fully represented, thereby enhancing the validity and reliability of subsequent system evaluations. Moreover, we preserved the distribution across POS tags as shown in Figure 5.2.

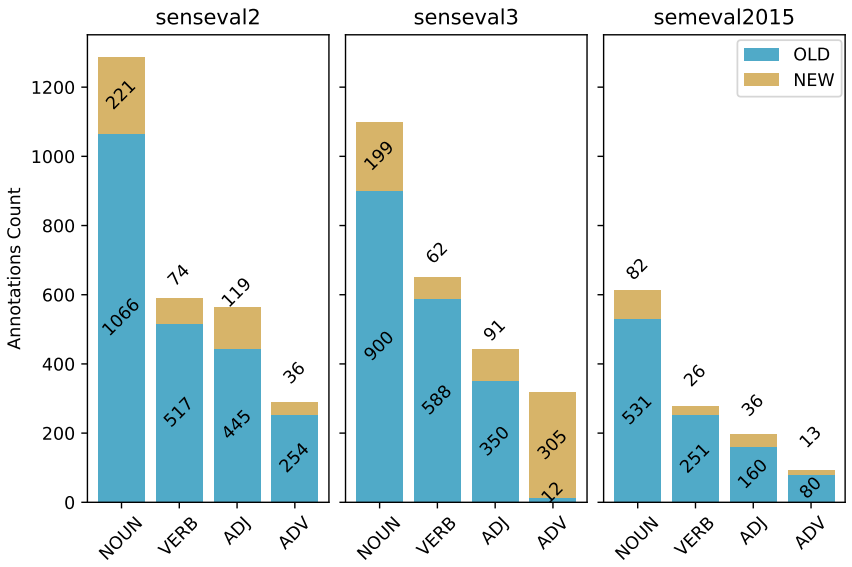

Figure 5.2. The counts of four POS categories (NOUN, VERB, ADJ, ADV) for three different datasets (Senseval2, Senseval3, Semeval2015). Each POS category is subdivided into 'OLD' (blue) and 'NEW' (orange) data points, indicating the frequency of each annotation before and after our comprehensive annotation process.

5.5.1 Word Sense Disambiguation

Setting We evaluate our model on all-word English WSD using WordNet as our sense inventory. WordNet provides a comprehensive and structured database of English word senses, making it a widely accepted benchmark for WSD. By utilizing WordNet, we ensure that our evaluation aligns with established standards in the field, facilitating comparison with previous work and other state-of-the-art models.

Comparison Systems We compare our model with recent state-of-the-art systems for WSD, which we divide into two different categories. On the one hand, we consider systems that frame WSD as a sequence-level classification problem, that is, systems that disambiguate a single span at a time. We include in our evaluation: Barba et al. (2021a, **ESCHER**), the first approach reformulating WSD as a text extraction problem; Zhang et al. (2022a, **KELESC**), a knowledge-enhanced version of ESCHER, incorporating additional information coming from WordNet; Song et al. (2021, **ESR**), an architecture framing WSD as a binary classification problem; and Barba et al. (2021c, **ConSeC**), the system that, thanks to iterative disambiguation, holds the state of the art on the reference benchmark at the time of writing.

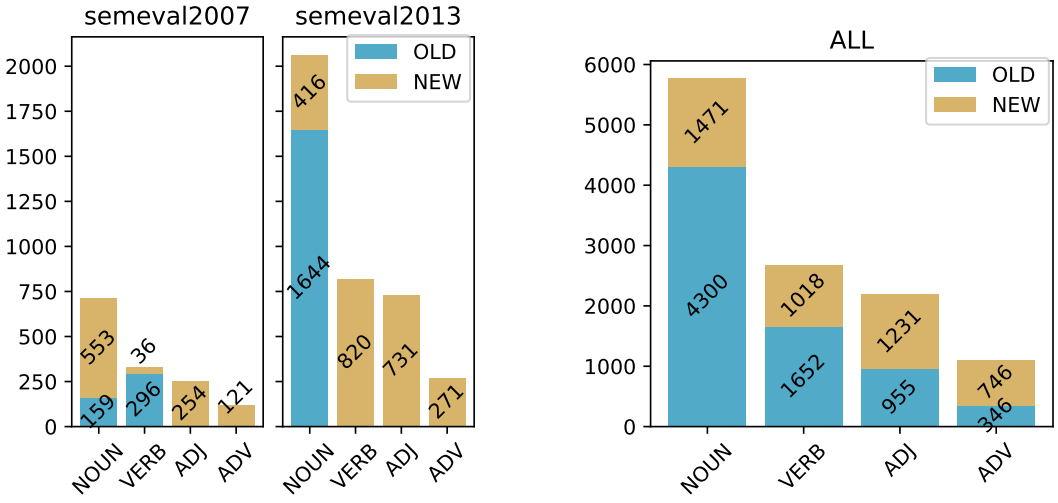

Figure 5.3. POS category counts before and after annotation. **Left:** Counts for the Semeval2007 and Semeval2013 datasets. The Semeval2007 dataset lacks annotations for ADJ and ADV categories, while Semeval2013 lacks annotations for VERB, ADJ, and ADV. The 'OLD' (blue) bars represent original counts, and the 'NEW' (orange) bars indicate counts post-annotation. **Right:** Counts for the 'ALL' dataset, which aggregates data across multiple sources. 'OLD' bars show original counts, and 'NEW' bars reflect increased counts after annotation.

On the other hand, we report models that frame WSD as a token-level classification problem, thus disambiguating all the spans in a sentence together. We include: Blevins and Zettlemoyer (2020, **BEM**), a bi-encoder system incorporating gloss knowledge; Bevilacqua and Navigli (2020, **EWISER**), a classifier modeling the relational knowledge in WordNet using a Personalized PageRank approach; and Conia and Navigli (2021, **WMLC**), a classification formulation for WSD that leverages the relational knowledge in WordNet at training time. We use this division to highlight not only the different designs of our comparison systems but, also, their corresponding trade-offs. Indeed, while sequence-level classifiers typically achieve higher performances, token-level classifiers emphasize speed and are considerably more usable in downstream applications.

Data We evaluate our model using the framework presented by Raganato et al. (2017). Specifically, we train our system on SemCor (Miller et al., 1993), perform model selection on SE07, and test on the other available datasets. We measure performances in terms of F1

	Models	Params	SE07	ALL	ALL_{FULL}
Sequence	ESCHER	400M	76.3	80.7	81.2
	KELESC	400M	76.7	81.2	81.4
	ESR	350M	77.0	81.1	81.3
	ConSeC	400M	**77.4**	**82.0**	**82.5**
Token	WMLC	340M	72.2	77.6	78.1
	EWISER	340M	71.0	78.3	78.9
	BEM	220M	74.5	79.0	79.7
	Our Model	295M	**75.2**	**80.2**	**80.8**

Table 5.2. WSD results for sequence-level and token-level classifiers.

score and, as in previous works, report this score on the concatenation of each evaluation dataset (identified as **ALL**). Finally, to set a reference for the WSL setting, we evaluate our model and the comparison systems on the novel benchmark presented in Section 5.4.2 under the WSD assumptions (identified as $\textbf{ALL}_{FULL}$).

Our Model Behavior Comparison systems operate in this setting assuming the availability of an oracle for both CD and CG. To enable a fair comparison, we

i) limit the candidates retrieved from the retriever module only to those that the spans to disambiguate can assume in WordNet and

ii) force the reader to select for each span only the senses that it can assume in WordNet.

Results Table 5.2 shows how all the systems under comparison fare on the standard framework. As a first result, we note that our model, a token-level classifier as all spans are disambiguated together, outperforms all the other token-level systems, almost reaching sequence-level ESCHER's performance. Furthermore, while the difference in comparison to ConSeC, i.e., the best-performing model, is significant in terms of F1 score (1.8 points), the speed comparison reveals a compelling advantage: using ALL as a reference, our model can process the 7253 instances in less than 17 seconds, while ConSeC requires a total of 73

seconds.[5] These results suggest that our model provides a solid foundation for our setting, showing not only strong performance on WSD, but also that it is flexible enough to support CD and CG. Finally, and most importantly, we can see that the performances obtained by the comparison systems on the $\mathbf{ALL}_{FULL}$ dataset align with those on the standard benchmark, allowing us to study how they change when relaxing the WSD constraints .

5.5.2 Word Sense Linking: Dropping the Concept Detection Oracle

Setting Here, we drop the R2 condition. In the context of WSL, this means that the system is required to identify spans of text that need to be disambiguated. By removing this oracle, we simulate a more challenging and practical setting where the system has to detect the spans to disambiguate before proceeding to candidate generation and disambiguation.

Comparison Systems Given the task's novelty, there are no direct comparison systems. We take advantage of this opportunity to assess the complexity of Concept Detection and its impact on ConSeC and BEM, that is, the best sequence-level and token-level classifiers described in the previous section. Specifically, to establish baselines, we implement two natural straightforward solutions for CD and pipeline each of these with our reference WSD systems. Our first solution consists of a simple heuristic approach: given an input text, we find all the longest spans such that the corresponding lemma is linked, through the mapping from word to possible senses, to any sense in WordNet. As an alternative to this strategy, we consider a supervised implementation to determine whether each token represents the start, inside, or outside of a span to disambiguate. We train[6] a variant of the architecture of Mueller et al. (2020) widely used in Named Entity Recognition tasks using BERT-base as an encoder. We denote the four variants by BEM$_{SUP}$, BEM$_{HEU}$, ConSeC$_{SUP}$ and ConSeC$_{HEU}$.

Data To address the issue of missing annotations in the training phase of our WSL system, specifically when using SemCor, which is known for its incomplete annotations, we propose a mitigation strategy. We use ConSeC$_{HEU}$ to identify and annotate all the missing spans in SemCor, leading to the creation of SemCor$_C$. This procedure results in 133 727 new

[5]To perform the comparison, we use the code made available by the authors at `https://github.com/SapienzaNLP/consec` using an RTX 3090 for both experiments.

[6]We use the span information contained in SemCor for training and SemEval-2007 for model selection.

	SE07			ALL_{FULL}		
Models	P	R	F1	P	R	F1
BEM_{SUP}	67.6	40.9	51.0	74.8	50.7	60.4
BEM_{HEU}	70.8	51.2	59.4	76.6	61.2	68.0
ConSeC_{SUP}	76.4	46.5	57.8	78.9	53.1	63.5
ConSeC_{HEU}	**76.7**	55.4	64.3	**80.4**	64.3	71.5
Our Model	73.8	**74.9**	**74.4**	75.2	**76.7**	**75.9**

Table 5.3. WSL results with no CD oracle.

Models	Lemmas	P	R	F1	Δ F1
ConSeC_{HEU}	all	80.4	64.3	71.5	–
ConSeC_{HEU}	one	71.6	56.4	63.1	-8.4
ConSeC_{HEU}	no	0.0	0.0	0.0	-71.5
Our Model	all	75.2	76.7	75.9	–
Our Model	one	70.4	73.1	71.7	-4.2
Our Model	no	68.5	62.5	65.4	-10.5

Table 5.4. WSL analysis on CG oracle.

annotations, in addition to the original 226 036 already in SemCor. We note that the usage of these annotations proves to be crucial for our model but irrelevant, if not actually harmful in some cases, for both BEM and ConSeC. Therefore, in what follows, the results we report always refer to the usage of SemCor_C for our model and of solely SemCor for our comparison systems.

Our Model Behavior Differently from the previous section, here we run the model in a WSL setting. The retriever identifies the sense candidates and the reader their corresponding spans. However, as the other systems use a CG oracle, for a fair comparison, for a given identified span, we limit the reader to selecting only the senses it can assume.

Results Table 5.3 reports the behavior of our model and the comparison systems. We note that all systems experience, in this unconstrained setting, a large drop in performance, with

scores diverging markedly from those reported for WSD in Table 5.2 on $\mathbf{ALL}_{FULL}$ (e.g., 82.5 vs 71.5 for ConSeC). All the same, our model behaves significantly better, attaining 75.9 (5 point drop) and surpassing the best alternatives, i.e., ConSeC$_{HEU}$, by almost 5 points. Interestingly, while our model does, in fact, lose to ConSeC$_{HEU}$ in terms of precision (80.4 vs 75.2), it displays large improvements in recall (64.3 vs 76.7).

To better understand these results, we investigate how CD approaches fare in terms of span recall.[7] What we find is that, while our model achieves 92.1, the two baselines lag behind by a considerable margin, with HEU and SUP reaching 80.6 and 68.2, respectively. Besides showing the effectiveness of our model on CD, this has several implications. First, tackling CD in a supervised fashion, in the current regime of WSL data, without taking into account the reference inventory, is not sufficient for good performance. Second, while HEU is a better baseline, neglecting the semantics, i.e. choosing the longest span with at least one valid sense, is deleterious. Paired together, these findings suggest that hybrid schemes like ours, where both training examples and the inventory are jointly employed, represent a better alternative. Finally, while HEU achieves nearly acceptable performance in English, the same does not hold for other languages where word inflection is a more complex phenomenon, and the mapping from word to senses lacks coverage compared to English[8]: for instance, testing span recall in Italian, using XL-WSD (Pasini et al., 2021), results in 70.7, 10 points less than English.

5.5.3 Word Sense Linking: Relaxing the Candidate Generation Oracle

Setting and Comparison Systems Here, we drop the R2 condition and examine different relaxations of R3. That is, compared to the setting of the previous section, we assume the mapping from word to possible senses, which we used up to this point as our Candidate Generation oracle, to be either incomplete or absent. Such scenarios are realistic as they mimic low-resource language settings in which dedicated WordNet and the related word-to-sense mappings are incomplete. We compare our model with ConSeC$_{HEU}$.

Data To highlight the importance of having a complete mapping between senses and their lemmas, we analyze the performance of both systems in three different settings, that is,

[7] This refers to the accuracy of systems in identifying spans in ALL, irrespective of their associated senses.

[8] This refers to the resources available to the research community.

when, for each synset, we have at our disposal:

i) all the lemmas with which it can be expressed,

ii) only its most frequent lemma, or

iii) no lemma at all.

In the first setting, where all possible lemmas are available, the system can utilize all the lexical variations for disambiguation. The second setting, which restricts the information to only the most frequent lemma, simulates a scenario with limited lexical resources, requiring the system to rely only on the most common representation of each sense. This may impact the system's ability to accurately disambiguate less frequent word forms. The third setting, where no lemma information is provided, forces the system to rely solely on the definition, and this is the most challenging scenario.

Our Model Behavior Our model behavior is unchanged compared to the previous section. The only difference across the three settings regards the textual representation of each sense in I, which now consists of its definition postpended to

i) all its lemmas,

ii) only its most frequent lemma, or

iii) no lemma at all.

Results As evidenced by Table 5.4, both systems perform worse when leveraging an incomplete mapping. However, our model shows a more robust behavior in such a setting compared to ConSeC$_{HEU}$. Indeed, as it performs the CG step independently from the mapping (with the retriever module), and takes advantage of the lemmas only to enhance sense representations, it is less reliant on this mapping. In contrast, ConSeC$_{HEU}$ requires the sense mapping in order to retrieve the candidates for the WSD step.

More in detail, when limiting to only the main lemma for each sense, ConSeC$_{HEU}$ can only predict senses that are expressed by their primary lemma. As expected, this results in a decrease of 8.4 F1 points. Our model, instead, is able to predict all senses, with a lower impact of 4.2 F1 points. Moving to the scenario where no lemmas are available, our system

is still able to recognize and disambiguate spans, but its performance drops significantly by 10.5 F1 points. Yet, compared with ConSeC$_{HEU}$, this is still a huge improvement: indeed, ConSeC$_{HEU}$ is completely unable to perform the task as it lacks candidates.

Our objective is to evaluate our models' effectiveness in scenarios with limited CG oracles, typical of most mid- and low-resource languages in WSD. The results we report not only underline the robustness of our model but also highlight the inadequacy of modern WSD systems.

5.6 Related Work

Since this work is the first to introduce Word Sense Linking, unsurprisingly, there is no previous literature that covers it. However, what is surprising is the total absence, to the best of our knowledge, of studies that examine how recent state-of-the-art WSD systems scale to real-world scenarios. This is especially puzzling when we consider the growing interest that WSD has been receiving. Indeed, thanks to the introduction of pretrained transformer-based language models (Devlin et al., 2019; Lewis et al., 2020; He et al., 2021b), this task has been witnessing renewed attention, with the research community focusing on challenging directions such as unseen prediction, cross-inventory generalization, and data efficiency, inter alia. Among the alternatives explored, the usage of definitions, which we also follow in this work, has proved to be particularly effective (Huang et al., 2019; Blevins and Zettlemoyer, 2020; Barba et al., 2021a), achieving unprecedented performances and allowing sense representations to be disentangled from their occurrences in the training corpus.

Yet, in spite of this trend and its promising results, to the best of our knowledge, no assessment of how these models might actually be applied in general real-world scenarios has been made, thereby overlooking the relevance of Concept Detection and Candidate Generation. We suspect that this oversight is due mainly to two reasons:

i) the performances on WSD benchmarks being too scarce, at least until recently, for any downstream application, and

ii) the research community assuming that adequate word sense mappings are always available for CG and that a simple heuristic approach could solve CD.

In our CD approach, we chose the greedy strategy, a decision supported by Martínez-Rodríguez et al. (2020), who observed its widespread use in span identification within texts. However, our study clearly shows that is not a trivial task. Both heuristic and supervised techniques report definitely suboptimal behaviors.

Finally, to provide some background on Candidate Generation, the task has generally been approached by striving to enumerate all the possible senses a word can assume. However, this is a prohibitively challenging endeavor, especially when wishing to scale across languages. While the research community has put forward a number of studies and mitigating strategies (Bikel, 2000; Al Tarouti and Kalita, 2016; Khodak et al., 2017; Neale, 2018), the resulting resources are still incomplete.

Arguably closest to our work, especially to our WSL model, is Zhang et al. (2022b), who address Entity Linking by initially generating candidates, followed by Mention Detection (akin to our Concept Detection) and Entity Disambiguation (similar to our Word Sense Disambiguation). However, their reader architecture differs significantly from ours: they link one candidate at a time, whereas our model can simultaneously process all spans in an input sequence. A similar retriever-reader approach is also explored in Orlando et al. (2024), where both Entity Linking and Relation Extraction are handled through a unified Retriever-Reader framework. Their model, like ours, retrieves candidate entities before jointly linking them in a single forward pass, improving efficiency compared to approaches that process candidates individually.

5.7 Comparison with EntQA

In this section, we present a detailed comparison between EntQA (Zhang et al., 2022b), and our model in the context of a WSL unconstrained environment. While EntQA represents a significant benchmark in the field, particularly for tasks similar to our WSL model, such as Entity Linking through the pipeline: candidate generation, Mention Detection (comparable to our Concept Detection), and Entity Disambiguation (analogous to our Word Sense Disambiguation), the two architectures diverge particularly in the reader part. Unlike EntQA, which processes candidates sequentially, our model can simultaneously handle all spans within an input sequence. We used the same Retriever for both reader models. In particular,

	SE07			ALL_{FULL}		
Models	P	R	F1	P	R	F1
ConSeC_{HEU}	**76.7**	55.4	64.3	**80.4**	64.3	71.5
EntQA	75.1	64.7	69.5	78.4	66.5	72.0
Our Model	73.8	**74.9**	**74.4**	75.2	**76.7**	**75.9**

Table 5.5. Our model comparison with EntQA in the WSL task tested on ALL_{FULL} dataset.

we re-implemented the EntQA reader model.

The performance outcomes, as shown in Table 5.5, show that our model outperforms EntQA. Our model obtained an F1 score of 75.9 compared to EntQA's 72.0. This superiority in performance underscores the effectiveness of our model, especially in terms of recall and overall F1 score. We argue that, in our model, contextualizing all the candidates of a sentence together plays a crucial role in the performance gain. Moreover, our model not only outperforms EntQA but it is also more efficient. By processing all candidates together, our model significantly reduces processing times, in contrast to EntQA's sequential candidate method. This efficiency is quantitatively evident as our system processes the ALL_{FULL} dataset in merely 17 seconds, a substantial improvement over the 63 seconds required by EntQA. To perform the comparison, we evaluated the same machine using an RTX 3090 for both experiments. This speed-up not only demonstrates our model's enhanced performance in terms of speed but also reinforces its practicality for integration into downstream tasks, further establishing our method's advantage in the field of WSL unconstrained settings.

5.8 Conclusion

In this chapter, we explored how integrating word and sentence representations can enhance semantic tasks, using Word Sense Linking as a case study. WSL extends traditional Word Sense Disambiguation by requiring systems to both identify relevant spans and disambiguate their meanings using a reference inventory. This approach aligns more closely with real-world applications, where predefined spans and sense mappings are not always available.

To tackle WSL, we introduced a retriever-reader architecture that effectively combines the strengths of sentence-level and word-level representations. This architecture allows

for a broader understanding of context through sentence embeddings, while simultaneously enabling precise sense disambiguation through word-level focus. Our experiments demonstrated that this combination improves upon the performances of classical approaches, highlighting the value of leveraging both granular word-level details and broader sentence-level context in semantic tasks.

Moreover, our analysis showed that while straightforward extensions of WSD systems to WSL often result in significant performance drops, our proposed model is more robust and consistently outperforms these extensions across different settings.

In the next chapter, we will build upon these insights by exploring how the latest advancements in Large Language Models can be leveraged for synthetic data generation. This approach aims to further improve the quality of representations, particularly in scenarios where data scarcity limits the effectiveness of traditional training methods.

∼ This page was intentionally left blank ∼

Chapter 6

Synthetic Data Generation to Address Data Scarcity

Abstract

In this chapter, we explore synthetic data generation as a solution to address data scarcity, particularly for domain-specific tasks where labeled data is limited or costly to obtain. Building on the integration of word- and sentence-level representations, we focus on generating synthetic data that can further refine representation learning. Leveraging Large Language Models (LLMs) for controlled data creation, we demonstrate their capacity to produce contextually accurate and diverse examples suited for tasks like factuality evaluation.

This chapter begins by examining the potential of LLMs in generating factual and unfactual data pairs, thereby supporting models trained to distinguish nuanced factual information. However, we address challenges like hallucination, a known issue in generative models, by implementing structured generation processes that anchor created data in verifiable knowledge. Specifically, we introduce a large-scale resource, LLM-OASIS, which combines factual-unfactual pairs derived from Wikipedia. Using this synthetic data, we train and benchmark models on tasks such as end-to-end factuality evaluation and evidence-based claim verification, achieving notable improvements.

Through experiments, we showcase how this approach enables smaller, specialized models to achieve comparable performance to larger models, underscoring the effectiveness of synthetic data in specialized representation learning.

6.1 Overview

In the previous chapter, we explored the integration of word-level and sentence-level representations, demonstrating their combined potential for tasks like Word Sense Linking. This highlighted how merging granular information with a broader contextual understanding can improve performance. Building on these insights, this chapter delves into the role of synthetic data generation in further refining representation learning, particularly in scenarios with limited labeled data or specific domain requirements.

In recent years, generative approaches in NLP have demonstrated remarkable results, achieving state-of-the-art performance across various tasks. This progress has been particularly notable with the advent of Large Language Models, which have revolutionized the field, driving advancements in many tasks, including Text Summarization (Goyal et al., 2022; Pu et al., 2023; Zhang et al., 2023b), Machine Translation (Alves et al., 2024; Zhang et al., 2023a; Wang et al., 2023), and Question Answering (Kamalloo et al., 2023; Rasool et al., 2024).

LLMs have also emerged as powerful tools for creating high-quality synthetic data (Brown et al., 2020a). These models can produce diverse and contextually accurate data that closely mimic real-world scenarios, providing a valuable resource when labeled data is scarce or challenging to obtain (Xie et al., 2020). This capability is particularly critical in domains like low-resource languages and specialized fields, where collecting extensive datasets is costly and time-consuming (Conneau and Lample, 2019).

However, LLMs also bring challenges, notably their tendency to produce hallucinations—statements that lack grounding in factual data (Tonmoy et al., 2024; Tam et al., 2022; Fadeeva et al., 2024). This issue is particularly problematic in factuality evaluation, where the goal is to ensure that the generated or retrieved content aligns with verifiable knowledge sources.

In this chapter, we focus on leveraging synthetic data creation to improve factuality evaluation systems. Synthetic data is particularly valuable when genuine labeled data is either sparse or entirely unavailable, as it allows models to be trained on data that closely mirrors the complexity of real-world scenarios. Unlike the uncontrolled hallucination problem, where LLMs may generate information that lacks grounding in verifiable sources, our approach to synthetic data creation is more controlled and structured. Specifically, we

use LLMs to generate pairs of ⟨factual, unfactual⟩ passages deliberately and transparently.

Furthermore, we continued the exploration of the retriever-reader paradigm (Chen et al., 2017). As highlighted in the previous chapter, this architecture excels in combining the strengths of sentence-level and word-level representations. Here, we extend its use to factuality evaluation, where the task requires retrieving passages that either support or refute a given claim. Unlike typical sentence retrieval models, which focus on semantic similarity (Reimers and Gurevych, 2019; Wang et al., 2022), our retriever is designed to locate passages that specifically confirm or contradict the provided claims, adding a layer of nuance to the retrieval process. The retrieved passages are then analyzed by a reader model using word-level detail to assess whether the evidence supports or contradicts the claim. This process mirrors our earlier application of the retriever-reader paradigm. Still, it adapts it to the unique challenges of factuality evaluation, where precise evidence retrieval and nuanced interpretation of the content are required.

The integration of synthetic data generation and the retriever-reader architecture provides several key advantages:

- It allows for the creation of training resources that are closely aligned with the specific needs of factuality evaluation, filling the gap where natural labeled data is unavailable or insufficient.

- The retriever-reader framework enables a seamless combination of sentence-level retrieval with word-level analysis, allowing models to leverage both context and detail when verifying information.

- Similar to its success in Word Sense Linking, this approach ensures that the models can identify relevant passages and interpret the fine-grained distinctions necessary for determining factuality.

However, while our approach seeks to address these challenges, existing resources have typically been designed with a narrower focus, targeting specific domains or tasks. For instance, several benchmarks have been developed for assessing factuality in specific contexts, such as news summarization (Laban et al., 2021; Tang et al., 2023), book content (Scirè et al., 2024), and dialogue systems (Tang et al., 2024). These benchmarks provide valuable insights but often lack the flexibility to be applied across different types of content. Similarly,

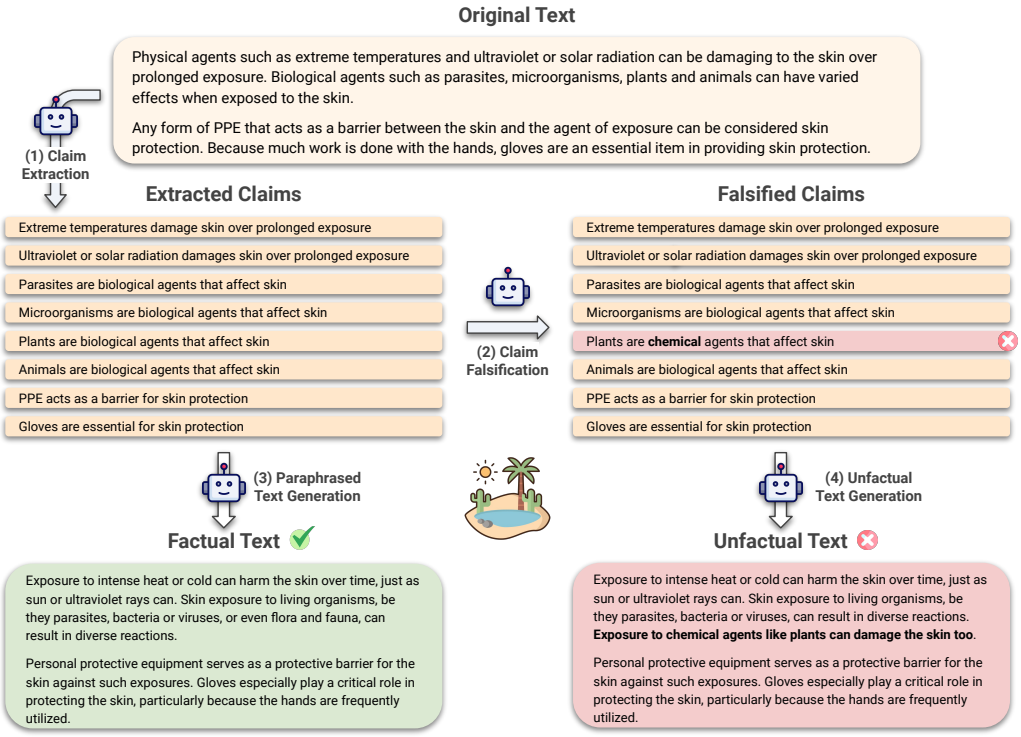

Figure 6.1. Pipeline for the creation of LLM-OASIS. Given a passage from a Wikipedia page (original text on top), we task an LLM to: extract a list of atomic claims (1), falsify one of the extracted claims (2), and then, given the two sets of claims, produce a paraphrase of the original text (3), and an alternative version featuring the unfactual information (4).

the FEVER benchmark (Thorne et al., 2018), which focuses on verifying individual claims against Wikipedia, is limited in its scope. It excels at validating isolated facts but struggles when applied to longer texts that contain multiple, interconnected pieces of information.

In this context, we introduce LLM-OASIS, a large-scale resource for end-to-end factuality evaluation, created by extracting and falsifying information from Wikipedia pages. The overall process is depicted in Fig. 6.1. As a result, we obtain 81k $\langle \text{factual}, \text{unfactual} \rangle$ pairs that are suitable to train end-to-end factuality evaluation systems.

Additionally, we set up a human annotation process to: i) create a gold standard for the factuality evaluation task that is useful for benchmarking LLMs and ii) validate the quality of the proposed data creation pipeline. Additionally, we issue two tasks, namely *end-to-end factuality evaluation* and *evidence-based claim verification* to benchmark LLMs. Our experiments demonstrate that our resource is challenging even for state-of-the-art models, both in zero-shot and Retrieval Augmented Generation (Lewis et al., 2021, RAG)

settings, with GPT-4o achieving an accuracy of 60% and 68%, respectively. In summary, our contributions are the following:

- We introduce (Scirè and Bejgu et al., 2024, LLM-OASIS), a large-scale synthetic dataset designed to improve representation learning for factuality evaluation. This resource is derived by systematically falsifying claims extracted from Wikipedia, creating a diverse set of $\langle \text{factual}, \text{unfactual} \rangle$ pairs that enhance models' ability to distinguish between truth and misinformation.

- Our dataset supports two key tasks: *end-to-end factuality evaluation* and *evidence-based claim verification*. These tasks leverage a retriever-reader architecture, where the retriever identifies passages that either support or contradict a claim using sentence-level representations, while the reader focuses on detailed word-level analysis to determine factual consistency.

- We develop a gold standard benchmark through a rigorous human annotation process, ensuring a reliable evaluation framework for testing the capacity of models in understanding nuanced representations of factuality. This benchmark aids in assessing how well models utilize the combined strengths of the retriever-reader approach.

- Our experiments demonstrate that LLM-OASIS presents a significant challenge for current LLMs, highlighting the need for better integration of word and sentence representations. When trained on synthetic data from LLM-OASIS, the retriever-reader architecture shows enhanced performance in understanding factual content, showcasing the value of targeted synthetic data in refining representation learning.

Although we selected Wikipedia as the basis for our resource, we emphasize that our methodology can be potentially adapted to any other corpus in any domain or language, as the only requirement is access to a collection of raw texts.

6.2 Related Work

Previous studies on factuality evaluation have focused on assessing *factual consistency*, or the extent to which a generated text is grounded in its source document. Resources for this task typically include human annotations indicating whether the generated text

accurately reflects the facts presented in the original document. However, many of these studies are tailored to specific tasks and domains, such as assessing factual consistency in news summaries (Fabbri et al., 2021; Tang et al., 2023; Pagnoni et al., 2021), books (Scirè et al., 2024), and dialogues (Tang et al., 2024).

Additionally, these studies often assume that the knowledge source required for verification is always available (e.g., the source document). This assumption does not hold in the broader task of *factuality evaluation*, where a text in natural language must be verified independently of the availability of supporting evidence, potentially requiring information retrieval techniques.

The first contribution towards general-purpose factuality evaluation dates back to FEVER (Thorne et al., 2018, Fact Extraction and VERification), which pairs claims with evidence retrieved from Wikipedia. The FEVER dataset comprises 185,445 human-generated claims, created by modifying sentences extracted from Wikipedia and subsequently verified without knowledge of the original sentences. The claims are classified as `Supported`, `Refuted`, or `NotEnoughInfo`, and for the first two categories, annotators also recorded the sentence(s) forming the necessary evidence for their judgment. Although challenging, FEVER presents limitations due to its focus on fact verification, which involves checking the veracity of individual claims. This focus is hardly adaptable to real-world scenarios, where texts to verify usually feature multiple claims. Additionally, FEVER's annotation effort is limited to a relatively-small subset of 10k English Wikipedia pages.

More recently, several studies have introduced strategies for generating synthetic instances for factuality evaluation. Notably, Muhlgay et al. (2024) introduced *FACTOR*, a framework for generating factuality benchmarks by prompting an LLM to produce both factual and unfactual completions given a prefix text. *FACTOR* includes 4,266 ⟨prefix, completion⟩ pairs, each annotated with a factuality label. Similarly, Chen et al. (2023) presented *FELM*, a benchmark consisting of 847 LLM outputs focused on various knowledge categories, including World Knowledge, Math, and Reasoning, with human-provided factuality annotations.

While these resources provide valuable benchmarks for evaluating the factuality of LLM outputs, their limited size constrains their applicability in training robust, generalizable factuality evaluators. This chapter addresses the need for expanded synthetic data generation

to create larger and more versatile datasets, specifically tailored to train models.

By leveraging the synthetic data generation approach introduced here, we aim to produce a resource that captures the complexity of real-world factuality evaluation across diverse contexts. This focus on synthetic data creation not only aligns with the objectives outlined in the introduction but also builds on the insights gained in previous chapters on representation learning, emphasizing the need for comprehensive, task-specific data to enhance model performance in factuality tasks.

With LLM-OASIS, we differentiate from previous studies by introducing a large-scale, task-agnostic resource covering a wide range of domains from Wikipedia. Specifically, LLM-OASIS enables the task of end-to-end factuality evaluation, namely the more realistic scenario that involves the verification of raw text in natural language. Notably, texts falling under this setting, always go beyond individual sentences, inherently posing a more complex challenge to the systems. Additionally, to the best of our knowledge, it is the largest resource for this task, featuring 162,550 passages in natural language and 681,201 claims, which can be verified against knowledge from 81,275 Wikipedia pages covering a broad set of domains. Finally, we reserve a manually curated subset for this task, consisting of approximately 2k instances, and use it to benchmark several state-of-the-art models.

6.3 LLM-OASIS

In this section, we outline the steps required to generate LLM-OASIS, a large-scale dataset for factuality evaluation, designed with a controlled synthetic data creation process. By grounding generation in factual passages from Wikipedia and guiding the LLM through specific, structured modifications rather than open-ended free-text generation, we aim to reduce the probability of hallucinations. This approach ensures that generated examples remain tied to verifiable data, thereby improving the reliability of training instances for factuality tasks.

We selected Wikipedia as our data source due to its comprehensive topic coverage and frequent updates, which support accuracy. To further enhance dataset quality, we retained only the most visited English Wikipedia pages.[1] Each page is segmented into passages of K

[1]We selected the 80k most popular pages from 2023.

sentences, organized with a sliding window and a stride of s sentences, forming our initial corpus.[2]

Following the workflow shown in Fig. 6.1, we use an LLM[3] to perform a structured, multi-step generation process:

- **Claim Extraction (Sec. 6.3.1):** The LLM identifies a set of atomic claims from each passage.
- **Claim Falsification (Sec. 6.3.2):** In a controlled step, one of the extracted claims is falsified.
- **Factual and Unfactual Text Generation (Sec. 6.3.3):** Using the extracted claims as a basis, the LLM produces two passages: a paraphrased factual version and an alternative version incorporating the falsified claim. This pairing of $\langle \text{factual}, \text{unfactual} \rangle$ passages yields reliable training examples aligned with verifiable knowledge.

This structured and grounded approach to synthetic data generation supports the creation of high-quality resources for factuality evaluation, addressing data scarcity while controlling for factual reliability. By anchoring the generation process in validated information and enforcing specific text modifications, our methodology mitigates hallucination risks and strengthens the development of robust factuality evaluators.

In the remainder of this section, for clarity, we describe the steps mentioned above individually. Still, we anticipate that the step-specific outputs will be obtained by means of a general, unified prompt containing instructions for all the steps. The overall prompt is provided in Table A.1.

6.3.1 Claim Extraction

The first step in creating LLM-OASIS involves extracting claims from an input passage t. We randomly sample one passage from each Wikipedia page and then extract a list of claims from each of the passages (cf. Step 1 in Fig. 6.1).

We frame the claim extraction task as an end-to-end autoregressive generation problem. Let $\mathcal{M}$ be our generative model. Given an input passage t, we task $\mathcal{M}$ to extract the

[2] In creating our resource, we set $K = 5$ and $s = 1$.

[3] We used the GPT-4 API. More details in Appendix A.3.

claims using the prompt $P_1(t)$ (cf. Step 1 in Table A.1):

$$\mathcal{M}(P_1(t)) = (c_1, \ldots, c_n) \tag{6.1}$$

where $(c_1, \ldots, c_n)$ represents the sequence of the generated claims. With the prompt $P_1(t)$, we aim at obtaining atomic[4] and self-contained claims, i.e. elementary units of information that do not require additional context to be verified. Specifically, we explicitly require the model to adhere to such formal definition, and, additionally, constrain it to generate short texts and avoid the usage of pronouns as subjects.
For instance, given the **input passage**:

> *"The Amazon Rainforest, also known as Amazonia, is a moist broadleaf forest in the Amazon biome that covers most of the Amazon basin of South America. This region includes territory belonging to nine nations, with Brazil containing 60% of the rainforest."*

the model $\mathcal{M}$ returns the following list of **claims**:

1. The Amazon Rainforest is also known as Amazonia.
2. It is a moist broadleaf forest in the Amazon biome.
3. The Amazon Rainforest covers most of the Amazon basin of South America.
4. The region includes territory belonging to nine nations.
5. Brazil contains 60% of the rainforest.

6.3.2 Claim Falsification

We introduce a critical factual error into one of the extracted claims to produce an unfactual version of the original text. Formally, given the set of claims $C = (c_1, \ldots, c_n)$, we task the model to falsify one of the claims as follows:

$$\mathcal{M}(P_2(C)) = (c_i, \bar{c}_i) \tag{6.2}$$

[4] Liu et al. (2023) defines a *claim* as an Atomic Content Unit (ACU), that is, an elementary unit of information found in a text that does not require further subdivision for the purpose of reducing ambiguity.

where $P_2(C)$ is the prompt comprising the instructions for claim falsification, $\bar{c}_i$ the falsified claim and c_i the corresponding factual one. We ask the model to provide the factual claim as well, thus enabling the investigation of the model's behavior.

As outlined in Table A.1 (Step 2), we instruct the model to falsify only one of the extracted claims by introducing a critical yet subtle error, which makes it potentially challenging to detect. Moreover, inspired by findings from previous works about the manual creation of Natural Language Inference (NLI) resources (Parrish et al., 2021; Hu et al., 2020), we designed the prompt with instructions to discourage the generation of naive contradicting instances, e.g., trivial negations of verbs. Continuing the example introduced in the previous section, given the extracted set of **claims**:

1. The Amazon Rainforest is also known as Amazonia.
2. The Amazon Rainforest is a moist broadleaf forest in the Amazon biome.
3. The Amazon Rainforest covers most of the Amazon basin of South America.
4. The region includes territory belonging to nine nations.
5. Brazil contains 60% of the rainforest. (c_i)

the model $\mathcal{M}$ produces the following **falsified claim**:

The majority of the forest is contained within Peru. ($\overline{c_i}$)

6.3.3 Factual and Unfactual Text Generation

Based on the claims extracted in the previous steps (cf. Sections 6.3.1 and 6.3.2), we now aim at generating pairs of ⟨factual, unfactual⟩ texts, which populate our resource for factuality evaluation, thus enabling the training and the benchmarking of factual reasoners.

Factual text generation To make the factuality evaluation task more challenging, we leverage paraphrase generation instead of using the original passages from Wikipedia as our factual texts. This approach produces texts that convey the same meaning as the original ones but with different surface forms, thereby making the verification task difficult for LLMs in both zero-shot settings – as the original texts could have been seen during pretraining –

and RAG settings, which might retrieve the exact passages from Wikipedia. Formally, given the set of extracted claims C, we task the model to generate a factual text $\mathcal{F}$ grounded on such claims:

$$\mathcal{M}(P_3(C)) = \mathcal{F}. \tag{6.3}$$

where $P_3(C)$ is the prompt with the instructions for obtaining a factual text through paraphrasing. As described in Table A.1 (Step 3) we explicitly require $\mathcal{M}$ to follow the sequence of extracted claims to encourage a full coverage of the facts expressed in the original text. For instance, given the following **claims**:

1. The Amazon Rainforest is also known as Amazonia.
2. The Amazon Rainforest is a moist broadleaf forest in the Amazon biome.
3. The Amazon Rainforest covers most of the Amazon basin of South America.
4. Brazil contains 60% of the rainforest.

the model $\mathcal{M}$ generates the following **factual text**:

> *"Amazonia, widely known as the Amazon Rainforest, is a damp broadleaf forest located within the Amazon biome, covering a significant portion of the Amazon basin in South America. This vast region spans across nine countries, with Brazil housing 60% of the rainforest."*

Unfactual text generation Finally, the unfactual texts are generated through an analogous process, this time grounded on the set of claims that includes the unfactual one, namely, $\overline{C} = (c_1, \ldots, \overline{c}_i, \ldots, c_n)$. We obtain the unfactual text $\mathcal{U}$ with the generation process defined with the following:

$$\mathcal{M}(P_4(\overline{C}, \mathcal{F})) = \mathcal{U} \tag{6.4}$$

where $P_4(\overline{C}, \mathcal{F})$ is the prompt containing the guidelines for unfactual text generation. In particular, as specified in Table A.1 (Step 4), we instruct $\mathcal{M}$ to produce a text identical to $\mathcal{F}$ except for the segment containing the factual error to ensure that the only confounding factor for the verification task is the unfactual portion of the text. This approach helps isolate

the effect of the factual inaccuracy, preventing the model to introduce further inaccuracies. For example, given the **claims** in $\overline{C}$:

1. The Amazon Rainforest is also known as Amazonia.
2. It is a moist broadleaf forest in the Amazon biome.
3. The Amazon Rainforest covers most of the Amazon basin of South America.
4. The region includes territory belonging to nine nations.
5. The majority of the forest is contained within Peru.

the model $\mathcal{M}$ generates the following **unfactual text**:

> *"Amazonia, widely known as the Amazon Rainforest, is a damp broadleaf forest located within the Amazon biome, covering a significant portion of the Amazon basin in South America. This vast region spans across nine countries, and the majority of the forest is cointained within Peru."*

6.4 The LLM-OASIS benchmark

As a result of the steps described in Section 6.3, we obtained a large resource consisting of claims and texts (both factual and unfactual) that can be used to train end-to-end factuality evaluation systems. However, due to the automated nature of the proposed approach, it is crucial to evaluate the quality of the produced data – by evaluating the individual steps of our pipeline – and introducing a gold-standard benchmark for the task.

6.4.1 Human Evaluation

To assess the quality of our dataset and enable a rigorous evaluation of our procedure, we asked $M = 5$ expert linguists to validate a portion of $N = 1,750$ instances for each task in our pipeline (cf. Sec. 6.3 and Fig. 6.1). Each annotator curated $(N/M) + K$ instances for each task with each of the M subsets having an overlap of $K = 100$ instances shared among all annotators. We paid the annotators according to the standard salaries for their geographical location and provided them with task-specific guidelines, annotation examples, and a simple interface for each task.

Claim Extraction	
# Pages	81,275
# Passages	81,275
Avg. Tokens per Passage	99.7
# Claims	681,201
Avg. Claims per Passage	8.381
Avg. Tokens per Claim	8.6
Claim Falsification	
# Unfactual Claims	81,275
Avg. Tokens per Unfactual Claim	9.0
Factual Text Generation	
# Factual Texts	81,275
Avg. Tokens per Factual Text	82.9
Unfactual Text Generation	
# Unfactual Texts	81,275
Avg. Tokens per Unfactual Text	86.5

Table 6.1. Summary statistics for the creation of LLM-OASIS.

Claim Extraction For the claim extraction task, annotators received Wikipedia passages $(t_1, \ldots, t_N)$, each accompanied by a list of claims extracted by the model $\mathcal{M}$ as described in Section 6.3.1. The annotators' task was to verify whether each claim was appropriately represented in the corresponding passage (i.e. with the same semantics) and assess their atomicity.[5]

We evaluated the LLM's performance on this task by counting the human-annotated errors, yielding an accuracy of 96.78%. Additionally, we measured inter-annotator agreement, resulting in a Fleiss' κ score of 0.81. These results underscore both the high quality

[5]We chose to prioritize a precision-oriented evaluation for two key reasons: First, low coverage does not affect our proposed claim verification task (see Task 2, Section 6.4.2); and second, evaluating coverage would have required annotators to read the entire passage, making the annotation process more time-consuming and costly.

Task	Accuracy (%)	Fleiss' κ
Claim Extraction	96.78	0.81
Claim Falsification	98.55	0.84
Factual Text Generation	90.36	0.73
Unfactual Text Generation	89.20	0.72

Table 6.2. Performance of the chosen LLM $\mathcal{M}$ in the data generation process according to human evaluation (Accuracy), and the corresponding inter-annotator agreement (Fleiss' κ).

of the generated $\langle$text, claims$\rangle$ pairs and the strong agreement among the annotators.

Claim Falsification For this task, annotators received pairs of claims $\langle c_i, \overline{c_i} \rangle$ with c_i being one of the original claims selected from $(c_1, \ldots, c_n)$ and $\overline{c_i}$ the corresponding falsified claim produced by the model $\mathcal{M}$. The annotators' task was to verify whether each claim was appropriately falsified (i.e. with contradicting semantics). This required them to determine if $\overline{c_i}$ meaningfully diverged from c_i in terms of content and truthfulness, effectively capturing the model's ability to produce altered, incorrect versions of the original claims. Again, the model achieved a very high accuracy (98.55%). We measured a Fleiss' κ score of 0.84, showing up almost perfect agreement between the annotators.

Factual and Unfactual Text Generation For these two tasks, we used a common format. Annotators received lists of original (or falsified) claims C (or $\overline{C}$) and the associated factual (or unfactual) texts produced by the model $\mathcal{M}$. The annotators' task was to verify whether each claim was correctly represented in the generated text. In the context of factual text generation, we additionally check whether the texts feature the same semantics as the claims but using a different wording. For the factual text generation step, we measured an accuracy of 90.36% and a Fleiss' κ score of 0.73. Similarly, for the unfactual text generation, we measured an accuracy of 89.2% and a Fleiss' κ score of 0.72.

Overall, the reported detailed evaluations summarized in Table 6.2 show the efficacy and robustness of the proposed methodology for producing training data for the task.

6.4.2 Gold Benchmark

In this section, we describe the construction of our benchmark, along with the factuality-oriented tasks we propose. Specifically, we exploit the human annotations (cf. Section 6.4.1) to construct a gold-standard benchmark for model evaluation. To ensure the high quality of our data, we only retain the instances that were not marked as error by any of our annotators in any annotation stage (cf. Sec. 6.4.1). We employ this data to propose the following two evaluation tasks, which we describe as follows.

Task 1: End-to-End Factuality Evaluation The first task is to determine whether a given text contains any factual inaccuracies. Formally, given an input passage t, the model must output a binary label $y \in \{\text{True}, \text{False}\}$, where True indicates that the text is factually accurate and False indicates the presence of factual inaccuracies.

For this setting, we rely only on factual and unfactual texts as input passages, and discard the original texts, as the latter might have already been seen during the pre-training of LLMs. Specifically, to further ensure the high quality of our benchmark, we only retain the correct paraphrases that are generated from a valid set of claims (cf. *Factual and Unfactual Text Generation* and *Claim Extraction* in Sec. 6.4.1). Concerning the valid unfactual texts, instead, we only keep the ones that are: i) generated, again, from a set of valid claims, and, ii) properly falsified and paraphrased (cf. *Claim Falsification* and *Factual and Unfactual Text Generation* in Sec. 6.4.1). We then labeled all the resulting factual and unfactual texts with *True*, and *False*, respectively.

In this setting, we aim at evaluating models on discerning true from fake texts (i.e., "Truth" from "Mirage"). This formulation enables the assessment of both plain LLMs and more complex RAG models. We deem this task to be particularly challenging as the falsification may involve even a single word occurring in one of the many claims featured in a text, in the spirit of recent works highlighting how LLMs struggle to deal with subtle nuances in a large input text (Kamradt, 2023; Hsieh et al., 2024; Laban et al., 2024; Wang et al., 2024a)

Task 2: Evidence-based Claim Verification In this setting, the task is to classify individual claims as factual or unfactual using a given evidence. This approach assumes that claims

are already extracted from the text, simplifying the task by focusing on isolated statements rather than the entire text to verify. Formally, given an input claim c and a corresponding evidence passage e, the model must output a binary label $y \in \{\text{True}, \text{False}\}$, where True indicates that the claim is supported by the evidence and False indicates that the claim is not supported by the evidence.

For this setting, we focus on the extracted claims and their corresponding unfactual version, and use the factual text as evidence. We discard both the original and unfactual texts as the former might have already been seen during the pre-training of LLMs, while the latter contradicts real-world knowledge and, therefore, the internal knowledge of LLMs, possibly leading to unfair evaluations.

Additionally, to guarantee the high precision of our data, we focus on the claims that are both atomic and reflecting the same semantics of the original text (cf. *Claim Extraction* Sec. 6.4.1). Then, we only keep the ones that have been appropriately falsified (cf. *Claim Falsification* in Sec. 6.4.1), along with their unfactual counterparts. Finally, we apply the same quality checks described in Task 1 to retain only the valid factual texts.

At this stage, we classify the $\langle c_i, \mathcal{F} \rangle$ pairs with the label True, while we label $\langle \overline{c_i}, \mathcal{F} \rangle$ as False, with c_i and $\overline{c_i}$ being the original claim and its falsified version, respectively.

6.5 End-to-end Factuality Evaluation with LLM-OASIS

In this section, we showcase how LLM-OASIS can be leveraged to build an end-to-end factuality evaluation system. In the spirit of Min et al. (2023c), we decompose the task of evaluating the factuality of a given text into three simpler tasks, namely, Claim Extraction, Evidence Retrieval and Claim Verification. The process begins with extracting a set of atomic facts (cf. **Claim Extraction**, Sec. 6.5.1) from the text to be verified. These extracted claims are then used to retrieve relevant evidence from a reliable knowledge base (cf. **Evidence Retrieval**, Sec. 6.5.2). After this, the factual accuracy of each claim is evaluated by comparing it against the retrieved evidence (cf. **Claim Verification**, Sec. 6.5.3). Finally, the results of these individual evaluations are aggregated to determine the overall factuality of the entire text.

6.5.1 Claim Extraction

Our approach starts by extracting atomic claims from a given input text t. With the aim of training a claim extractor, we leverage LLM-OASIS to create a dataset of $\langle t, C \rangle$ tuples, where t is an original text from Wikipedia and $C = (c_1, ..., c_n)$ the corresponding automatically-extracted claims by our chosen LLM $\mathcal{M}$ (Section 3.1). We then fine-tune a smaller sequence-to-sequence model $\mathcal{G}$ on this data, thus distilling the claim extraction capabilities of $\mathcal{M}$.

We frame the training process as a text generation task; more formally, we fine-tune $\mathcal{G}$ to generate the claims autoregressively:

$$P(y \mid t) = \prod_{k=1}^{|y|} P\left(y_k \mid y_{0:k-1}, t\right) \tag{6.5}$$

where y is the sequence obtained by concatenating the claims in C and y_k is a token in this sequence.

6.5.2 Evidence Retrieval

At this stage, given the claims extracted by $\mathcal{G}$, we require a strategy for retrieving relevant passages from a knowledge corpus to serve as evidence to verify those claims. Again, we leverage LLM-OASIS to create our training dataset; in particular, given each generic claim $c_j \in C$ extracted from the original text t, we construct the following training pairs:

$$\langle c_j, t \rangle, \langle c_j, \mathcal{F} \rangle, \langle c_j, \mathcal{U} \rangle, \forall c_j \in C$$

where $\mathcal{U}$ and $\mathcal{F}$ are the generated factual and unfactual texts (cf. Section 6.3.3).

We then augment this set by pairing the factual and unfactual texts with the falsified claim $\overline{c_i}$ (cf. Section 6.3.2), thus obtaining the following additional training instances:

$$\langle \overline{c_i}, t \rangle, \langle \overline{c_i}, \mathcal{F} \rangle, \langle \overline{c_i}, \mathcal{U} \rangle.$$

In this way, we include all possible pairs of $\langle$claim, passage$\rangle$ in LLM-OASIS in our training set. This strategy is aimed at increasing the generalization capabilities of our retriever: notably, given a claim, the retriever is trained to both provide the passages to support it along with the ones that are useful to negate it.

Following the methodology outlined in DPR (Karpukhin et al., 2020), we define our retriever $\mathcal{E}$ as a Transformer-based encoder, which produces dense representations of both claims and passages. Starting from an input claim c and a knowledge corpus $\mathcal{D}$, we use $\mathcal{E}$ to compute a vector representation v_c for c, and v_p for every passage $\{p_1, p_2, \ldots, p_m\} \in \mathcal{D}$. Then, we use the dot product $v_c \cdot v_p$ to rank all the passages in $\mathcal{D}$ and, finally, extract the top k among these. The resulting k passages form our evidence set $R_k(c, \mathcal{D})$ for c.

Finally, we minimize the DPR loss $\mathcal{L}$ to train $\mathcal{E}$:

$$\mathcal{L} = -\sum_{i=1}^{N} \log \frac{e^{v_{c_i} \cdot v_{p_i^+}}}{e^{v_{c_i} \cdot v_{p_i^+}} + \sum_{j \neq i} e^{v_{c_i} \cdot v_{p_j^-}}} \tag{6.6}$$

where N is the batch size, v_{c_i} is the vector representation of the i-th claim in the batch, $v_{p_i^+}$ is the vector representation of the corresponding gold passage for the i-th claim, and $v_{p_j^-}$ represents the vector representations of all the other passages in the batch, serving as in-batch negatives. This formulation ensures that the model learns to score the correct passage higher than the other ones within each batch, which has been shown to be an effective strategy for training retrieval models (Yih et al., 2011; Gillick et al., 2019).

6.5.3 Claim Verification

The final step of our factuality evaluation methodology involves verifying each claim c generated by our claim extractor from the text t, by comparing it against the corresponding passages $R_k(c, \mathcal{D})$ retrieved from our corpus. Inspired by previous work on consistency evaluation (Zha et al., 2023; Scirè et al., 2024; Chen and Eger, 2023), we ground our verification approach on the NLI formulation. NLI is a task that determines the logical relationship between two texts: a *premise* and a *hypothesis*. Formally, given a premise pre and a hypothesis hyp: $\text{NLI}(pre, hyp) = Y \in \{\text{ENT}, \text{NEUT}, \text{CONTR}\}$, where Y is a label indicating whether pre entails (ENT), is neutral about (NEUT), or contradicts (CONTR) hyp.

Training a claim verifier on LLM-Oasis In this section, we show how LLM-OASIS can be leveraged to train a model for the claim verification task. Complying to the NLI formulation, we require a strategy to assess whether each claim extracted from a text is entailed, contradicted, or neutral with respect to a set of the retrieved passages. With this purpose, we construct a training dataset by deriving the following ⟨claim, passage, label⟩ triplets from

LLM-OASIS:

$$\langle c_j, t, \text{ENT}\rangle, \langle c_j, \mathcal{F}, \text{ENT}\rangle, \forall c_j \in C$$

where $c_j \in C$ is a claim extracted by the LLM from the original text t (cf. Section 6.3.1), and $\mathcal{F}$ and $\mathcal{U}$ are the factual and unfactual texts outlined in Section 6.3.3.

We expand our training dataset for NLI with the following triplets:

$$\langle \overline{c_i}, t, \text{CONTR}\rangle, \langle \overline{c_i}, \mathcal{F}, \text{CONTR}\rangle, \langle \overline{c_i}, \mathcal{U}, \text{ENT}\rangle, \langle c_i, \mathcal{U}, \text{CONTR}\rangle$$

where $\overline{c}_i$ is the falsified version of the extracted claim c_i (cf. Sec. 6.3.2).

To obtain a complete NLI dataset, we require a strategy to generate neutral triplets as well. To achieve this, we first pair each claim c_j in C (Section 6.3.1) with the passages p_i of the Wikipedia page W from where the original text t was extracted. Then, we select the passage p^* as the one that maximizes the neutrality probability when fed to an NLI model[6] Ψ along with c_j:

$$p^* = \underset{p_i \in W}{\text{argmax}}\, \mathbb{P}_\Psi(\text{NEUT} \mid p_i, c_j)$$

and augment our dataset with the neutral pairs $\langle c, p^*, \text{NEUT}\rangle$. This approach increases the likelihood that the selected passages are semantically related to the claim, as they come from the same Wikipedia page, while still being neutral. This is preferable to randomly selecting neutral examples, as it tends to provide more meaningful contrasts.

Finally, we fine-tune a Cross-Encoder model on this data; as a result of this process, we obtained our claim verification model Φ. More information about the training setup can be found in Section 6.6.1.

Claim verification algorithm In Alg. 1 we outline how we leverage Φ to assess the factuality of a claim. Our procedure takes as input a claim, a set of *top-k* retrieved passages, and a claim verification model. For each ⟨passage, claim⟩ pair we obtain a label $\hat{y}$ by applying Φ:

$$\hat{y} = \Phi(p_i, c) = \underset{y \in \{\text{ENT}, \text{NEUT}, \text{CONTR}\}}{\text{argmax}} P(y \mid p_i, c) \tag{6.7}$$

[6]We used a DeBERTa-v3-large model fine-tuned on several NLI datasets. For more information: `https://huggingface.co/MoritzLaurer/DeBERTa-v3-large-mnli-fever-anli-ling-wanli`

Algorithm 1 Algorithm for Claim Verification.

Require: claim c, top-k retrieved passages $\{p_1, p_2, \ldots, p_k\}$, NLI model Φ

for each passage p_i in $\{p_1, p_2, \ldots, p_k\}$ **do**
  $\hat{y} \leftarrow \Phi(c, p_i)$
  if $\hat{y} ==$ ENT **then**
    return True
  else if $\hat{y} ==$ CONTR **then**
    return False
  end if
  {The output of the model is NEUT, i.e., neutrality. Continue to the next passage}
end for
return True {All NLI outputs are neutral, c is deemed verified}

where p_i is a retrieved passage and c is a claim, which are fed to the NLI model Φ as the premise and hypothesis, respectively. As described in Alg. 1, the algorithm proceeds by checking the output of this model for each passage in the ranking order. If Φ outputs ENT for a passage, the claim is deemed verified (i.e., return True). Conversely, if Φ outputs CONTR, the claim is deemed unfactual (i.e., return False). Finally, if Φ outputs NEUT for all passages, the claim is deemed verified (i.e., return True), as there is no contradicting evidence available.

In practice, given an input text t, we used our claim verifier to assign a factuality label to the claims generated by our claim extractor, using the passages returned by our retriever as evidences.

The final factuality prediction for the text t is an aggregation of the claim-level factuality labels. Specifically, the text t is considered factual if all of its extracted claims are verified, unfactual otherwise.

6.6 Experimental Setup

In this section, we provide details about the models and data involved in our experiments. To train our components for the end-to-end factuality evaluation task, we leverage the synthetic data from LLM-OASIS (cf. Section 6.3, Figure 6.1). Specifically, we randomly split the

passages in a 80/20 proportion to build the train and validation datasets, respectively. When splitting, we ensure that all the claims, the factual and the unfactual text generated from the same passage will end up in the same split.

We evaluate both our modular architecture (cf. Sec. 6.6.1) and several LLM-based baselines (cf. Sec. 6.6.2), showing the effectiveness of our benchmark in challenging factuality evaluation systems. To assess their performance, we rely on the LLM-OASIS gold-standard benchmark (Section 6.4.2). Models are evaluated across the two proposed tasks (i.e. *end-to-end verification* and *evidence-based claim verification*), and we use balanced accuracy (Brodersen et al., 2010) as our evaluation metric.[7]

6.6.1 Our model

Here, we provide the training details for each module of our proposed solution for end-to-end factuality evaluation (cf. Sec. 6.5).

Claim extractor As described in Section 6.5.1, we build our claim extractor dataset with the ⟨text, claims⟩ tuples in the training split of LLM-OASIS. We split the resulting dataset into ∼67k passage-claims pairs for training, and ∼4k passage-claims pairs for validation. Statistics about the claim extraction dataset can be found in Table 6.1.

We fine-tune a $T5_{base}$ (Raffel et al., 2019) model on this data to generate the sequence of claims given an input passage. We train the model for a total of 1M steps, with Adafactor (Shazeer and Stern, 2018) as optimizer with a learning rate of $1e^{-}5$.

Following Scirè et al. (2024), we rely on the easiness_{F1} metric for model selection. Let C represent the set of generated claims and C^* the set of gold claims. To compute the easiness_P score, as defined by Zhang and Bansal (2021), we first calculate the ROUGE-1[8] score for each generated claim $c \in C$ by comparing it to every gold claim $c^* \in C^*$, and then select the maximum score. The final easiness_P score is obtained by averaging these maximum scores over all generated claims:

$$\text{easiness}_P(C, C^*) = \frac{\sum_{c \in C} \max_{c^* \in C^*} \text{R1}(c, c^*)}{|C|} \tag{6.8}$$

[7] All our experiments are carried out on a single *NVIDIA GeForce RTX 3090* GPU.

[8] We consider ROUGE-1 to be a suitable basis for our easiness metric due to the high extractiveness of the claim extraction task.

Similarly, we compute the easiness$_R$ score by selecting the maximum ROUGE-1 score for each gold claim c^* with respect to all generated claims c

$$\text{easiness}_R(C, C^*) = \frac{\sum_{c^* \in C^*} \max_{c \in C} \text{R1}(c, c^*)}{|C^*|} \tag{6.9}$$

Finally, we combine easiness$_P$ and easiness$_R$ to calculate the easiness$_{F1}$ score, and select the model that achieved the highest easiness$_{F1}$ on our validation set.

Evidence Retriever The training dataset of our retriever comprises ∼3.2M ⟨claim-evidence⟩ pairs. At validation/test time, we construct the knowledge corpus with the original texts in our validation split and gold benchmark, respectively. We expand the corpus with passages from the same Wikipedia page to make the evaluation more realistic and challenging. This approach results in our corpus $\mathcal{D}$ comprising a total of 2.5M passages.

We use the pre-trained Transformer-based architecture $E5_{base}$ (Wang et al., 2022) as our encoder $\mathcal{E}$. To generate embeddings for both claims and passages, we apply mean pooling over the output of $\mathcal{E}$. The model is trained with a batch size of 20 input texts for 300 000 steps, using AdamW (Loshchilov and Hutter, 2019) as the optimizer. We employ a learning rate of $1 \cdot 10^{-6}$, with a 20% warm-up phase.

Claim Verifier As outlined in Section 6.5.3, we formalize the claim verification task as an NLI problem and construct a dataset of ∼3.5M ⟨premise, hypothesis, label⟩ triplets from LLM-OASIS. We devoted 3.2M instances for training our claim verification model and the remaining 300k for validation. We fine-tune DeBERTa-v3$_{large}$ (He et al., 2021c) for a total of 1M steps on this data, using Adafactor.

6.6.2 Baselines

We comprehensively evaluate a set of LLM-based baselines on the LLM-OASIS benchmark. We compare closed-source models from the GPT family (OpenAI et al., 2024), including the latest GPT-4o, to open-source LLMs such as Llama 3 (Touvron et al., 2023) and Mistral (Jiang et al., 2023a).[9] As part of the end-to-end task evaluation, we ablate the impact of

[9]Due to our computational constraints, we opt to focus our evaluation of open-source LLMs on models of up to 8B parameters.

providing the LLMs with external knowledge, i.e., in the RAG setting. To experiment this, we include the top-K passages[10] returned by our retriever (cf. Section 6.5.2) in the prompts. Details about the utilized prompts and parameters for the end-to-end factuality evaluation and evidence-based claim verification tasks are provided in Appendix A.2 and A.3, respectively.

6.7 Task 1: End-to-End Factuality Evaluation

This section presents the experimental setup and results for the end-to-end factuality evaluation task, which relies on synthetic data to train and test the evidence retrieval and factuality verification modules. Synthetic data is essential in this context as it provides controlled training resources that align with the specific requirements of factuality assessment. Without fine-tuning, sentence retrieval models typically perform poorly on this task, as they are not designed to identify passages that confirm or refute factual claims. This controlled synthetic data generation addresses these limitations by equipping the models with task-specific knowledge that improves retrieval precision and verification accuracy.

Evidence Retrieval The evidence retrieval module is a critical component as it supplies the external knowledge required for the claim verification process. The quality of retrieved evidence directly impacts the end-to-end factuality evaluation, establishing an upper bound on external knowledge integration. We evaluate evidence retrieval using the Recall at k (R@k) metric, defined as:

$$\text{R@k} = \frac{|\{\text{relevant } D\} \cap \{\text{top } k \text{ retrieved } D\}|}{|\{\text{relevant } D\}|} \tag{6.10}$$

This metric assesses our retriever's ability to locate relevant passages for factual verification among the top k ranked results. We set $k = 30$ for our experiments, balancing performance and efficiency, as shown in Figure 6.2. The fine-tuned $E5_{base}$ model achieved a Recall@30 (R@30) of 0.95, a substantial improvement over the model without fine-tuning, which only reached an R@30 of 0.52. This underscores the importance of the pretraining and

[10]We selected K=30 based on the analysis of our retriever's performance at different values of K (cf. Sec. 6.7) conducted on the validation set.

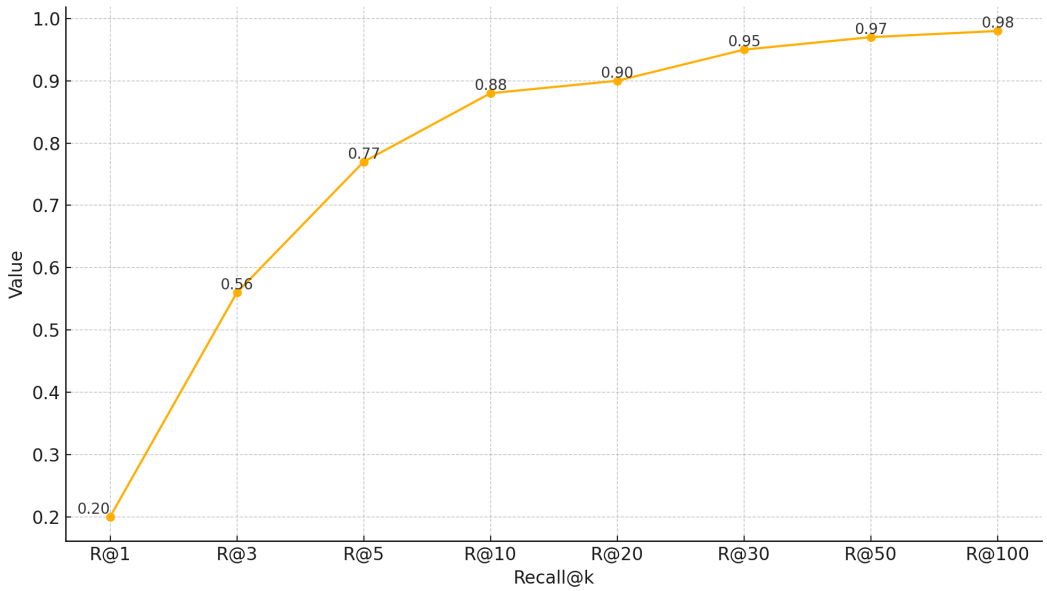

Figure 6.2. Recall@k performance of the $E5_{base}$ model at different values of k.

fine-tuning process on 3.2M passages, proving that synthetic data generation is crucial to achieving high retrieval accuracy for factuality tasks.

End-to-End Factuality Evaluation The results for the end-to-end factuality evaluation task are presented in Table 6.3. The balanced accuracy scores of the Large Language Models (LLMs) without external knowledge are relatively low, only marginally surpassing the random baseline. This outcome highlights the challenging nature of our benchmark, as the LLMs must verify factuality without external evidence, relying solely on prior knowledge, which proves insufficient for nuanced factual distinctions.

Incorporating external knowledge from our retriever module significantly improves accuracy, particularly for GPT-4o (+8%). In this setup, our model achieves a balanced accuracy of 0.69, outperforming all evaluated LLMs and illustrating the benefits of task-specific fine-tuning on synthetic data. Despite having fewer parameters (1B) than its competitors (7B+), our model's performance demonstrates that high-quality synthetic data can effectively compensate for a smaller model size, optimizing both resource use and accuracy. However, even the best-performing system achieves only $\sim$ 70 in balanced accuracy, underscoring the complexity of the task and the potential for further research in this domain.

Model	Params	B-Accuracy	
		w/o ext.	w/ ext.
Llama 3	7B	0.53	0.54
Mistral	7B	0.51	0.51
GPT-3.5	-	0.51	0.53
GPT-4o	-	0.60	0.68
Our Model	**1B**	-	**0.69**

Table 6.3. Balanced accuracy (B-Accuracy) of different models for end-to-end factuality evaluation. "Params" denotes the number of parameters in billions (B). "w/o" stands for without external knowledge and "w/" stands for with external knowledge.

Further Details on Evidence Retrieval As demonstrated in previous chapters, fine-tuning sentence representations for task-specific needs is essential. In Chapter 4, we highlighted the benefits of task-specific tuning for sentence alignment, where fine-tuned sentence embeddings improved alignment precision by focusing on cross-lingual semantic coherence. Similarly, in Chapter 5, task-adapted representations proved instrumental in enhancing Word Sense Linking by ensuring nuanced disambiguation. These examples underline the importance of tailoring sentence embeddings to specific tasks, a concept we expand upon in this chapter.

However, a persistent challenge in achieving such specialized representations is the availability of high-quality, task-specific labeled data. For many nuanced tasks, especially those like factuality evaluation that require verification of specific information, annotated data is either limited or completely absent. Acquiring such data through manual annotation is costly and time-intensive, particularly for domains with complex verification requirements or for under-resourced languages and domains.

To address this, synthetic data generation emerges as a valuable solution. By leveraging generative models to create large volumes of annotated data, we can systematically cover a wide range of scenarios and linguistic constructs relevant to factuality evaluation. This approach allows us to generate data that mirrors the characteristics of real-world information verification tasks, filling critical gaps left by the scarcity of naturally labeled datasets.

Model	Recall@30
$E5_{base}$ (without fine-tuning)	0.52
$E5_{base}$	0.95
bert-base-uncased	0.85
$E5_{small}$	0.75
$E5_{large}$	0.96

Table 6.4. Performance of Different Models on Claim Retrieval Task

Synthetic data generation, therefore, not only accelerates the fine-tuning process but also enhances the robustness and generalizability of the representations, enabling sentence retrieval models to perform better on specialized verification tasks.

Here, we examine the effectiveness of our evidence retriever for factuality verification through an ablation study, summarized in Table 6.4. The $E5_{base}$ model, after fine-tuning on synthetic data generated explicitly for factuality verification, achieved a Recall@30 of 0.95, compared to only 0.52 without fine-tuning. This substantial increase underscores the necessity of task-specific pretraining, which enables the model to adapt to the intricacies of factuality tasks, identifying passages that confirm or refute claims rather than simply matching for similarity.

We also explored various model architectures within the $E5$ family to balance computational efficiency with retrieval performance. While the $E5_{large}$ model achieved a marginally higher Recall@30 of 0.96, the performance gain was offset by significantly increased computational demands. Consequently, $E5_{base}$ was selected as the optimal model for our final system due to its efficiency.

This ablation study confirms that synthetic data generation for task-specific fine-tuning is essential. By training sentence retrieval models on data that captures the complexity of factuality tasks, we enable these models to go beyond general similarity measures. This is particularly crucial for factuality verification, where distinguishing between factual and unfactual information requires contextual alignment and precise evidence-based verification.

Model	# Parameters	B-Accuracy
Llama 3	7B	0.76
Mistral	7B	0.64
GPT-3.5	–	0.75
GPT-4o	–	0.90
Our Model	**0.4B**	**0.93**

Table 6.5. Balanced accuracy (B-Accuracy) of different models for evidence-based claim verification. "Params" denotes the number of parameters in billions (B).

6.8 Task 2: Evidence-based Claim Verification

In this section, we present the results for the second task we aim to evaluate with our benchmark (see Section 6.4.2), i.e., evidence-based claim verification. In this setting, we remove the claim extraction and retrieval modules, thus directly providing the claim verifier (cf. Section 6.5.3) with the ⟨evidence, claim⟩ pairs in our benchmark. For the LLM baselines, we used the same prompt as for the end-to-end setting, but here, we replaced the retrieved external passages with the gold evidence. Again, as shown in Table 6.5, our small specialized system outperforms all its competitors including GPT-4o (0.93 vs 0.9). Notably, all systems achieve higher performance compared to the previous setting, e.g., our system goes from 0.69 in the end-to-end task to 0.93. We attribute this to three main factors. First, this task is a simpler instance of the previous one, namely, the model is required to verify a single claim rather than a passage. Second, the system is provided with the exact evidence needed to verify the claim, while in the end-to-end formulation, each model relies on several passages returned by the retriever, hence possibly introducing noise in the process. Finally, the end-to-end verification implies reading and reasoning on a huge context (4k tokens on average) rather than the limited one (100 tokens on average) of this task.

6.9 Conclusion

In this chapter, we explored the utility of synthetic data generation as a powerful means to enhance representation learning across NLP tasks, particularly in scenarios where labeled

data is sparse or domain-specific. By creating controlled and well-defined synthetic datasets, we can fine-tune representations that meet the unique demands of specialized tasks. This approach not only mitigates the limitations of general-purpose embeddings but also allows for tailored adaptations that align closely with complex evaluation criteria.

Specifically, we focused on factuality evaluation, introducing LLM-OASIS, a large-scale dataset constructed from pairs of $\langle \text{factual}, \text{unfactual} \rangle$ passages based on Wikipedia content. LLM-OASIS enables robust training for factuality evaluation systems by providing a nuanced, context-rich source of synthetic data that mimics real-world scenarios. This synthetic resource is designed to refine both retrieval and interpretive accuracy, equipping models with the ability to discern between factual and unfactual statements more effectively.

Our experiments showed that: i) Large Language Models, whether in zero-shot or Retrieval-Augmented Generation settings, face challenges on this benchmark, emphasizing the value of task-specific tuning in factuality evaluation; and ii) smaller specialized models trained on LLM-OASIS achieved competitive performance, underscoring the importance of synthetic data in enhancing model capabilities for this task. These findings support the broader thesis objective of using synthetic data to create more precise and adaptable representations tailored to specific NLP challenges.

∼ This page was intentionally left blank ∼

Chapter 7

Conclusions

7.1 Summary of Contributions

This thesis presents a comprehensive exploration into the limitations of general-purpose embeddings in domain-specific NLP tasks and proposes a framework for specialized representation learning. Recognizing the critical need for adaptable embeddings in low-resource and complex domains, we developed methodologies that integrate word-level and sentence-level representations while introducing synthetic data generation as a solution to data scarcity. Our main contributions are as follows:

- **Enhanced Sentence and Word Representations:** We explored traditional word embedding approaches, leading to advanced sentence-level embedding. By leveraging the nuances of contextual dependencies and structural sensitivity, we improved sentence alignment and semantic similarity tasks.
- **Merging Word- and Sentence-level Representations:** By integrating granular, word-level information with holistic sentence-level embeddings, we showed the advantages of a unified approach. This combined representation outperformed independent methods in tasks requiring both precision and context, such as Word Sense Linking.
- **Synthetic Data for Specialized Embedding Tuning:** Recognizing the scarcity of annotated data for domain-specific tasks, we developed a synthetic data generation pipeline. Utilizing Large Language Models, we created LLM-OASIS, a large-scale resource designed to improve factuality evaluation. This dataset provides a reliable

benchmark for training models in fact verification, especially in complex, real-world scenarios.

7.2 Reflections on Synthetic Data and Embedding Adaptation

Throughout the chapters, we have observed that manually curated data often leads to the highest performance across NLP tasks. Such datasets, crafted with domain-specific accuracy and clear task alignment, enable models to capture nuanced meanings, contextually appropriate semantics, and fine-grained distinctions crucial for advanced tasks like word sense linking and sentence alignment. However, developing these datasets is costly, time-intensive, and frequently infeasible, especially in low-resource domains or underrepresented languages.

As Large Language Models (LLMs) have advanced, they offer promising solutions for data generation, capable of producing high-quality synthetic data that can supplement or even replace some aspects of manual curation. This thesis explores how carefully designed synthetic data can approximate the effectiveness of manual datasets. We systematically generated controlled pairs of factual and unfactual data, as well as targeted embedding adaptations, to address some of the inherent limitations in NLP—such as the challenge of accurate fact verification across diverse contexts. These synthetic datasets enable more resource-efficient, domain-sensitive models without the requirement of extensive human annotation.

By designing task-specific embeddings and focusing on each task's unique requirements, we demonstrated that even smaller, more specialized models can approach, or sometimes match, the performance of larger, general-purpose models. For instance, adapting embeddings to meet the needs of factuality evaluation and sentence alignment led to robust models with high accuracy, underscoring that effective model performance can be achieved with task-specific data resources rather than solely relying on increased model size.

This thesis thus provides evidence that embedding adaptation—supported by targeted, high-quality synthetic data—can substantially improve model performance across varied NLP tasks. The findings suggest a promising direction for future research in developing versatile, efficient models for specialized applications, paving the way toward more practical,

accessible NLP systems that retain precision and interpretability without the full dependency on large-scale, manually curated datasets.

7.3 Challenges and Limitations

While our approaches improved performance across tasks, several challenges remain. The dependency on LLMs for synthetic data generation raises concerns regarding hallucination and knowledge gaps. Though we mitigated these issues through structured prompts and careful verification, completely eliminating such model biases remains a challenge. Additionally, the reliance on Wikipedia as a source limits the scope of generalization across domains. Future work could explore multi-domain synthetic datasets or apply our methods to other corpora.

Our approach also demonstrates that task-specific fine-tuning can yield significant gains in representation learning. However, these findings may vary when applied to languages and domains with even fewer resources than we tested. Extending this work to cover such languages would require additional techniques to ensure effective cross-lingual transfer and resource-efficient training.

7.4 Concluding Remarks

In summary, this thesis introduces a multi-faceted approach to address the limitations of general-purpose embeddings for specialized NLP tasks. By integrating diverse representation strategies and leveraging synthetic data generation, we showed that high-performance, task-specific embeddings are achievable without large-scale resources. This work lays the groundwork for future advancements in creating adaptable, interpretable, and domain-sensitive embeddings, ultimately moving towards NLP systems capable of tackling an array of specialized tasks with precision and reliability.

∼ This page was intentionally left blank ∼

Part II

Appendices

Synthetic Data Generation to Address Data Scarcity

A.1 LLM-OASIS data generation prompt

In this section, we provide a description of the steps used to generate our resource, referencing Table A.1 for the unified prompt containing instructions for all the steps.

First, STEP 1: Claim Extraction involves extracting atomic claims from the provided text. An atomic claim is a semantically coherent piece of text that requires no further subdivision and must be short, using 15 words at most. Each claim must follow the logical flow of the original text and should avoid the usage of pronouns. All claims present in the input text must be reported in the list. This step ensures that we have a granular and clear representation of the original information.

Next, STEP 2: Factual Text Generation involves generating a paraphrased version of the original text once the atomic claims are extracted. The paraphrase must overlap as little as possible with the original text while preserving the meaning and following the same logical flow as the extracted claims. This step is essential for creating a version of the text that maintains the integrity of the information but is distinct in wording from the original. It ensures diversity in expression while keeping the core information intact.

Then, STEP 3: Claim Falsification involves subtly altering one of the extracted claims to introduce a critical factual inaccuracy. The altered claim must be relevant to the input text and should avoid naive negative transformations of verbs (e.g., was -> was not, did -> did not). This step returns the set of unaltered claims along with the altered one. The purpose of this task is to create a controlled instance of misinformation that can be used to

Input: Wikipedia Passage of K sentences

Instructions: Execute the following steps:

Step 1 - Claim extraction: From the input passage, extract a comprehensive set of claims. These claims must be atomic, i.e. semantically-coherent pieces of text that do not require further subdivision, and self-contained, i.e. not requiring additional context to be verified. Note that each claim must be short, using 15 words at most. Do not use "..." to truncate them. The ordering of the extracted claims must follow the logical flow expressed in the original text. Use a noun as the subject in the claim (avoid pronouns). All the claims that are featured in the input text must be reported in the list.

Step 2 - Claim falsification: From the output of Step 1, subtly alter one claim, in order to introduce a critical factual inaccuracy. Such claim must be the most relevant for the input text. It is forbidden to change dates, years, numbers and person/location/organization/etc. names. It is also forbidden to provide naive negative transformations of verbs, e.g., was -> was not, did -> did not. This step, i.e., Step 2, returns a pair containing the altered claim along with the original one.

Step 3 - Factual text generation: From the output of Step 1, generate a text. Note that this text must be a paraphrase of the original provided text, i.e. a new text that should overlap as little as possible with the original, while preserving the meaning. The generated text must follow the same logical flow as the ordering of the extracted claims.

Step 4 - Unfactual text generation: Generate a text from the final set of claims (original unaltered + altered) i.e. the output of Step 3. Note that the output of this step is not the original text, but the one generated from the final set of claims. Therefore this text contains unfactual information. The generated text must follow the same logical flow as the ordering of the claims. The output text must be as similar as possible to the output of Step 2, unless the unfactual part.

Output format: Return the output in a JSON with the following format: { 'step_1': List[str], 'step_2': Tuple[str, str], 'step_3': str, 'step_4': str}. The output must be a valid JSON, thus try to avoid special characters like ' and " inside the JSON values, unless you escape them with a \. Do not include any marker for the altered claim inside the JSON values, e.g., # this is the altered claim. Please do not provide any preamble to your response, just give me the JSON.

Table A.1. Prompt for the generation of data in LLM-OASIS.

study its effects.

Finally, STEP 4: Unfactual Text Generation involves generating a new text from the final set of claims (original unaltered plus the altered one). The generated text should not be the original text but one created from the final set of claims, thus containing unfactual information. The new text must follow the same logical flow as the ordering of the claims and should be as similar as possible to the factual text from Step 2, except for the introduced inaccuracy. This step produces a text that incorporates the misinformation, providing a basis for analyzing the impact of the altered claim.

The above-mentioned steps are executed using a general, unified prompt containing the instructions for all the steps. This approach ensures that the language model processes the tasks in a coherent and integrated manner. The detailed instructions for each step are provided in Table A.1.

A.2 Prompts for End-to-End Factuality Evaluation

To accomplish the task of end-to-end factuality evaluation, we employ different prompting strategies depending on the language model being used. For models like Llama, which supports a system prompt, we set specific instructions as the system message. For models like Mistral, which do not support a system prompt, we include the instructions at the beginning of the text. In our experiments, we set the temperature to 0.7 to control the randomness of the generated outputs, ensuring a balance between diversity and relevance. Other hyperparameters include a maximum token length of 5 and a beam search with a width of 5 for decoding the outputs. These settings were chosen to optimize the model's performance while maintaining computational efficiency.

The unified prompt used for factuality evaluation is provided in Table A.2.

The system message is set as follows:

"You are a highly-accurate fact-checker. Your task is to determine the factuality of a given text. A text is considered 'Factual' only if it is completely factually-accurate and contains no factual inaccuracies. Even a single small factual inaccuracy should result in a 'Not Factual' determination. Answer with just 'Factual' or 'Not Factual' without any explanation."

To further evaluate the impact of external knowledge, we prompted the LLMs with the same pieces of evidence retrieved and used by our NLI module (cf. Chap. 6). The updated prompt is shown in Table A.3. The same prompt was also used for the experiment in Sec. 6.8 for the Evidence-based claim verification. Moreover, we updated the system prompt with the following:

"You are a highly-accurate fact-checker. Your task is to determine the factuality of a given text using the evidence provided. A text is considered 'Factual' only if it is completely factually accurate and contains no factual inaccuracies. Even a single small factual inaccuracy should result in a 'Not Factual' determination. If evidence is not available, use your prior knowledge to make the assessment. Answer with just 'Factual' or 'Not Factual' without any explanation."

We tested different prompts and found that this one led to the best results. By providing clear examples and detailed instructions, we aim to ensure that the model accurately assesses the factuality of the given texts. This structured approach helps in training and evaluating the models effectively, ensuring high accuracy in end-to-end factuality evaluation.

A.3 Details about the employed LLMs

In this section, we detail the models we used in this work. For the generation of our dataset, we used GPT-4 API, with an approximate cost of $2000. As for the open-source models we utilized for the LLM baselines, we used the instruction-tuned versions of Llama 3 [1] and Mistral [2] publicly available on Hugging Face. For the benchmark evaluation, we utilized the OpenAI API. Specifically, for GPT-4o, we employed the model *GPT-4o-2024-05-13*. For GPT-3.5, we used *GPT-3.5-turbo-16k-0613* due to the necessity of handling a task with context exceeding the maximum window size of the standard GPT-3.5 model. For the claim-extractor, we use the pre-trained T5-base [3] as our base model.

[1] https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.3

[2] https://huggingface.co/meta-Llama/Meta-Llama-3-8B-Instruct

[3] https://huggingface.co/google-t5/t5-base

Determine whether the given text is factual or not.

Input Text:

INPUT_TEXT

Instructions:

1. Read the input text.

2. Evaluate the factual accuracy of the input text based on your training data and knowledge.

3. If the input text is factually-accurate, i.e. supported by known information, respond with "Factual."

4. If the input text contains any factual inaccuracies, i.e., it is not supported by known information, respond with "Not Factual."

5. Do not generate any additional text to the answer.

Examples:

Example 1:

Input Text: "The Eiffel Tower is located in Berlin."

Output: "Not Factual"

Example 2:

Input Text: "The Eiffel Tower is located in Paris."

Output: "Factual"

Example 3:

Input Text: "The Great Wall of China is visible from the moon."

Output: "Not Factual"

Example 4:

Input Text: "The Great Wall of China is a historical structure built in ancient China."

Output: "Factual"

Table A.2. Prompt for factuality evaluation of a text.

Determine whether the given text is factual or not using the evidence. If the information is not present in the evidence, rely on prior knowledge.

Input Text:

INPUT_TEXT

Evidence:

EVIDENCE

Instructions:

1. Read the input text.

2. Read the evidence if provided.

3. Assess whether the input text is factual based on the evidence if present.

4. If the evidence are not provided or is insufficient, use your prior knowledge to determine the factuality.

5. Respond with "Factual" if the input text is supported by evidence or prior knowledge.

6. Respond with "Not Factual" if the input text contains inaccuracies.

Examples:

Example 1:

Input Text: "The Eiffel Tower is located in Berlin."

Evidence: "The Eiffel Tower is located in Paris."

Output: "Not Factual"

Example 2:

Input Text: "The Eiffel Tower is located in Paris."

Evidence: "The Eiffel Tower is located in Paris."

Output: "Factual"

Example 3:

Input Text: "The Earth orbits the sun."

Evidence: "The Earth is round."

Output: "Factual"

Table A.3. Prompt for text factuality evaluation integrating external knowledge.

Bibliography

Sadaf Abdul-Rauf, Mark Fishel, Patrik Lambert, Sandra Noubours, and Rico Sennrich. 2012. Extrinsic evaluation of sentence alignment systems. In *Workshop on Creating Cross-language Resources for Disconnected Languages and Styles*, pages 6–10.

Feras Al Tarouti and Jugal Kalita. 2016. Enhancing automatic Wordnet construction using word embeddings. In *Proceedings of the Workshop on Multilingual and Cross-lingual Methods in NLP*, pages 30–34, San Diego, California. Association for Computational Linguistics.

Felipe Almeida and Geraldo Xexéo. 2023. Word embeddings: A survey.

Duarte M. Alves, José Pombal, Nuno M. Guerreiro, Pedro H. Martins, João Alves, Amin Farajian, Ben Peters, Ricardo Rei, Patrick Fernandes, Sweta Agrawal, Pierre Colombo, José G. C. de Souza, and André F. T. Martins. 2024. Tower: An open multilingual large language model for translation-related tasks.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2018. A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 789–798, Melbourne, Australia. Association for Computational Linguistics.

Mikel Artetxe and Holger Schwenk. 2019. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *Transactions of the Association for Computational Linguistics*, 7:597–610.

Claire Barale, Mark Klaisoongnoen, Pasquale Minervini, Michael Rovatsos, and Nehal

Bhuta. 2023. AsyLex: A dataset for legal language processing of refugee claims. In *Proceedings of the Natural Legal Language Processing Workshop 2023*, pages 244–257, Singapore. Association for Computational Linguistics.

Edoardo Barba, Tommaso Pasini, and Roberto Navigli. 2021a. ESC: Redesigning WSD with extractive sense comprehension. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4661–4672, Online. Association for Computational Linguistics.

Edoardo Barba, Luigi Procopio, Niccolo Campolungo, Tommaso Pasini, and Roberto Navigli. 2021b. Mulan: Multilingual label propagation for word sense disambiguation. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pages 3837–3844.

Edoardo Barba, Luigi Procopio, and Roberto Navigli. 2021c. ConSeC: Word sense disambiguation as continuous sense comprehension. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1492–1503, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Regina Barzilay and Lillian Lee. 2003. Learning to paraphrase: An unsupervised approach using multiple-sequence alignment. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 16–23.

Andrei Bejgu, Edoardo Barba, Luigi Procopio, Alberte Fernández-Castro, and Roberto Navigli. 2024. Word sense linking: Disambiguating outside the sandbox. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 14332–14347, Bangkok, Thailand. Association for Computational Linguistics.

Michele Bevilacqua and Roberto Navigli. 2020. Breaking through the 80% glass ceiling: Raising the state of the art in word sense disambiguation by incorporating knowledge graph information. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2854–2864, Online. Association for Computational Linguistics.

Michele Bevilacqua, Tommaso Pasini, Alessandro Raganato, and Roberto Navigli. 2021. Recent trends in word sense disambiguation: A survey. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021*, pages 4330–4338. ijcai.org.

Daniel M. Bikel. 2000. Automatic WordNet mapping using word sense disambiguation. In *2000 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 142–147, Hong Kong, China. Association for Computational Linguistics.

Terra Blevins and Luke Zettlemoyer. 2020. Moving down the long tail of word sense disambiguation with gloss informed bi-encoders. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1006–1017, Online. Association for Computational Linguistics.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Kay Henning Brodersen, Cheng Soon Ong, Klaas Enno Stephan, and Joachim M. Buhmann. 2010. The balanced accuracy and its posterior distribution. In *2010 20th International Conference on Pattern Recognition*, pages 3121–3124.

Samuel Broscheit. 2019. Investigating entity knowledge in BERT with simple neural end-to-end entity linking. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 677–685, Hong Kong, China. Association for Computational Linguistics.

Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

Peter F. Brown, Jennifer C. Lai, and Robert L. Mercer. 1991. Aligning sentences in parallel corpora. In *29th Annual Meeting of the Association for Computational Linguistics*, pages 169–176, Berkeley, California, USA. Association for Computational Linguistics.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020a. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020b. Language models are few-shot learners.

Mehmet Talha Cakmak, Süleyman Acar, and Gülsen Eryigit. 2012. Word alignment for english-turkish language pair. In *LREC*, pages 2177–2180.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1870–1879, Vancouver, Canada. Association for Computational Linguistics.

Shiqi Chen, Yiran Zhao, Jinghan Zhang, I-Chun Chern, Siyang Gao, Pengfei Liu, and Junxian He. 2023. Felm: Benchmarking factuality evaluation of large language models.

Stanley F. Chen. 1993. Aligning sentences in bilingual corpora using lexical information. In *31st Annual Meeting of the Association for Computational Linguistics*, pages 9–16, Columbus, Ohio, USA. Association for Computational Linguistics.

Yanran Chen and Steffen Eger. 2023. MENLI: Robust evaluation metrics from natural language inference. *Transactions of the Association for Computational Linguistics*, 11:804–825.

Stephane Clinchant, Kweon Woo Jung, and Vassilina Nikoulina. 2019. On the use of BERT for neural machine translation. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 108–117, Hong Kong. Association for Computational Linguistics.

Simone Conia and Roberto Navigli. 2021. Framing word sense disambiguation as a multi-label problem for model-agnostic knowledge integration. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3269–3275, Online. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Angel Daza and Anette Frank. 2020. X-SRL: A parallel cross-lingual semantic role labeling dataset. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3904–3914, Online. Association for Computational Linguistics.

Vincent J Della Pietra. 1994. The mathematics of statistical machine translation: Parameter estimation. *Using Large Corpora*, page 223.

Claudio Delli Bovi, Luca Telesca, and Roberto Navigli. 2015. Large-scale information extraction from textual definitions through deep syntactic and semantic analysis. *Transactions of the Association for Computational Linguistics*, 3:529–543.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of*

the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Bosheng Ding, Chengwei Qin, Ruochen Zhao, Tianze Luo, Xinze Li, Guizhen Chen, Wenhan Xia, Junjie Hu, Anh Tuan Luu, and Shafiq Joty. 2024. Data augmentation using LLMs: Data perspectives, learning paradigms and challenges. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 1679–1705, Bangkok, Thailand and virtual meeting. Association for Computational Linguistics.

Chris Dyer, Victor Chahuneau, and Noah A Smith. 2013. A simple, fast, and effective reparameterization of ibm model 2. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 644–648.

Philip Edmonds and Scott Cotton. 2001. SENSEVAL-2: Overview. In *Proceedings of SENSEVAL-2 Second International Workshop on Evaluating Word Sense Disambiguation Systems*, pages 1–5, Toulouse, France. Association for Computational Linguistics.

Kawin Ethayarajh. 2019. How contextual are contextualized word representations? Comparing the geometry of BERT, ELMo, and GPT-2 embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 55–65, Hong Kong, China. Association for Computational Linguistics.

Alexander R. Fabbri, Wojciech Kryściński, Bryan McCann, Caiming Xiong, Richard Socher, and Dragomir Radev. 2021. Summeval: Re-evaluating summarization evaluation.

Ekaterina Fadeeva, Aleksandr Rubashevskii, Artem Shelmanov, Sergey Petrakov, Haonan Li, Hamdy Mubarak, Evgenii Tsymbalov, Gleb Kuzmin, Alexander Panchenko, Timothy Baldwin, Preslav Nakov, and Maxim Panov. 2024. Fact-checking the output of large language models via token-level uncertainty quantification. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 9367–9385, Bangkok, Thailand. Association for Computational Linguistics.

Fangxiaoyu Feng, Yinfei Yang, Daniel Cer, Naveen Arivazhagan, and Wei Wang. 2022. Language-agnostic BERT sentence embedding. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 878–891, Dublin, Ireland. Association for Computational Linguistics.

Alexander Fraser and Daniel Marcu. 2007. Measuring word alignment quality for statistical machine translation. *Computational Linguistics*, 33(3):293–303.

Viviana Gaballo. 2012. Exploring the boundaries of transcreation in specialized translation. *ESP Across Cultures*, 9(1):95–113.

William A. Gale and Kenneth W. Church. 1993. A program for aligning sentences in bilingual corpora. *Computational Linguistics*, 19(1):75–102.

William A Gale, Kenneth W Church, and David Yarowsky. 1992. A method for disambiguating word senses in a large corpus. *Computers and the Humanities*, 26(5):415–439.

Qin Gao and Stephan Vogel. 2008. Parallel implementations of word alignment tool. In *Software engineering, testing, and quality assurance for natural language processing*, pages 49–57.

Sarthak Garg, Stephan Peitz, Udhyakumar Nallasamy, and Matthias Paulik. 2019. Jointly learning to align and translate with transformer models. *arXiv preprint arXiv:1909.02074*.

Daniel Gillick, Sayali Kulkarni, Larry Lansing, Alessandro Presta, Jason Baldridge, Eugene Ie, and Diego Garcia-Olano. 2019. Learning dense representations for entity retrieval. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 528–537, Hong Kong, China. Association for Computational Linguistics.

Luís Gomes and Gabriel Pereira Lopes. 2016. First steps towards coverage-based sentence alignment. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 2228–2231, Portorož, Slovenia. European Language Resources Association (ELRA).

Tanya Goyal, Junyi Li, and Greg Durrett. 2022. News summarization and evaluation in the era of gpt-3.

Joao Graca, Joana Paulo Pardal, Luísa Coheur, and Diamantino Caseiro. 2008. Building a golden collection of parallel multi-language word alignment. In *LREC*.

Zhiyu Guo and Minh Le Nguyen. 2020. Document-level neural machine translation using BERT as context encoder. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing: Student Research Workshop*, pages 101–107, Suzhou, China. Association for Computational Linguistics.

Zellig S. Harris. 1954. Distributional structure.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021a. Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021b. Deberta: decoding-enhanced bert with disentangled attention. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021c. Deberta: Decoding-enhanced bert with disentangled attention. In *International Conference on Learning Representations*.

Mickel Hoang, Oskar Alija Bihorac, and Jacobo Rouces. 2019. Aspect-based sentiment analysis using BERT. In *Proceedings of the 22nd Nordic Conference on Computational Linguistics*, pages 187–196, Turku, Finland. Linköping University Electronic Press.

Maria Holmqvist and Lars Ahrenberg. 2011. A gold standard for english-swedish word alignment. In *Proceedings of the 18th Nordic conference of computational linguistics (NODALIDA 2011)*, pages 106–113.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. 2024. Ruler: What's the real context size of your long-context language models?

Hai Hu, Kyle Richardson, Liang Xu, Lu Li, Sandra Kübler, and Lawrence S. Moss. 2020. OCNLI: original chinese natural language inference. *CoRR*, abs/2010.05444.

Luyao Huang, Chi Sun, Xipeng Qiu, and Xuanjing Huang. 2019. GlossBERT: BERT for word sense disambiguation with gloss knowledge. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3509–3514, Hong Kong, China. Association for Computational Linguistics.

Ignacio Iacobacci, Mohammad Taher Pilehvar, and Roberto Navigli. 2015. SensEmbed: Learning sense embeddings for word and relational similarity. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 95–105, Beijing, China. Association for Computational Linguistics.

Vivek Iyer, Edoardo Barba, Alexandra Birch, Jeff Pan, and Roberto Navigli. 2023. Code-switching with word senses for pretraining in neural machine translation. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 12889–12901, Singapore. Association for Computational Linguistics.

Masoud Jalili Sabet, Philipp Dufter, François Yvon, and Hinrich Schütze. 2020. SimAlign: High quality word alignments without parallel training data using static and contextualized embeddings. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1627–1643, Online. Association for Computational Linguistics.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023a. Mistral 7b.

Chao Jiang, Mounica Maddela, Wuwei Lan, Yang Zhong, and Wei Xu. 2020. Neural crf model for sentence alignment in text simplification. *arXiv preprint arXiv:2005.02324*.

Ting Jiang, Shaohan Huang, Zhongzhi Luan, Deqing Wang, and Fuzhen Zhuang. 2023b. Scaling sentence embeddings with large language models. *ArXiv*, abs/2307.16645.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547.

Ehsan Kamalloo, Nouha Dziri, Charles Clarke, and Davood Rafiei. 2023. Evaluating open-domain question answering in the era of large language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5591–5606, Toronto, Canada. Association for Computational Linguistics.

Gregory Kamradt. 2023. Needleinahaystack.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Martin Kay and Martin Röscheisen. 1993. Text-translation alignment. *Comput. Linguistics*, 19:121–142.

Mikhail Khodak, Andrej Risteski, Christiane Fellbaum, and Sanjeev Arora. 2017. Automated WordNet construction using word embeddings. In *Proceedings of the 1st Workshop on Sense, Concept and Entity Representations and their Applications*, pages 12–23, Valencia, Spain. Association for Computational Linguistics.

Philipp Koehn. 2004. Pharaoh: A beam search decoder for phrase-based statistical machine translation models. In *Machine Translation: From Real Users to Research*, pages 115–124, Berlin, Heidelberg. Springer Berlin Heidelberg.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of machine translation summit x: papers*, pages 79–86.

Philippe Laban, Alexander R. Fabbri, Caiming Xiong, and Chien-Sheng Wu. 2024. Summary of a haystack: A challenge to long-context llms and rag systems.

Philippe Laban, Tobias Schnabel, Paul N. Bennett, and Marti A. Hearst. 2021. Summac: Re-visiting nli-based models for inconsistency detection in summarization.

Patrik Lambert, Adrià De Gispert, Rafael Banchs, and José B Mariño. 2005. Guidelines for word alignment evaluation and manual alignment. *Language Resources and Evaluation*, 39(4):267–285.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2021. Retrieval-augmented generation for knowledge-intensive nlp tasks.

Jing Li, Aixin Sun, Jianglei Han, and Chenliang Li. 2022. A survey on deep learning for named entity recognition. *IEEE Transactions on Knowledge and Data Engineering*, 34(1):50–70.

Frederick Liu, Han Lu, and Graham Neubig. 2018. Handling homographs in neural machine translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1336–1345, New Orleans, Louisiana. Association for Computational Linguistics.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3730–3740, Hong Kong, China. Association for Computational Linguistics.

Yang Liu and Maosong Sun. 2015. Contrastive unsupervised word alignment with non-local features. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29.

Yixin Liu, Alexander R. Fabbri, Pengfei Liu, Yilun Zhao, Linyong Nan, Ruilin Han, Simeng Han, Shafiq Joty, Chien-Sheng Wu, Caiming Xiong, and Dragomir Radev. 2023.

Revisiting the gold standard: Grounding summarization evaluation with robust human evaluation.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Lieve Macken. 2010. An annotation scheme and gold standard for dutch-english word alignment. In *7th conference on International Language Resources and Evaluation (LREC 2010)*, pages 3369–3374. European Language Resources Association (ELRA).

David Mareček. 2008. Automatic alignment of tectogrammatical trees from czech-english parallel corpus. Master's thesis, Charles University, MFF UK.

Federico Martelli, Andrei Stefan Bejgu, Cesare Campagnano11, Jaka Čibej14, Rute Costa10, Apolonija Gantar14, Jelena Kallas, Svetla Koeva, Kristina Koppel, Simon Krek, et al. 2023. Xl-wa: a gold evaluation benchmark for word alignment in 14 language pairs.

Federico Martelli, Roberto Navigli, Simon Krek, Jelena Kallas, Polona Gantar, Svetla Koeva, Sanni Nimb, Bolette Sandford Pedersen, Sussi Olsen, Margit Langemets, Kristina Koppel, Tiuu Üksik, Jaka Dobrovoljc, Rafael-J. Ureña-Ruiz, José-Luis Sancho-Sánchez, Veronika Lipp, Tamás Váradi, András Győrffy, Simon László, Valeria Quochi, Monica Monachini, Francesca Frontini, Carole Tiberius, Rob Tempelaars, Rute Costa, Ana Salgado, Jaka Čibej, and Tina Munda. 2021. Designing the ELEXIS parallel sense-annotated dataset in 10 european languages. In *Proceedings of the eLex Conference*.

José-Lázaro Martínez-Rodríguez, Aidan Hogan, and I. Lopez-Arevalo. 2020. Information extraction meets the semantic web: A survey. *Semantic Web*, 11:255–335.

Rada Mihalcea and Ted Pedersen. 2003. An evaluation exercise for word alignment. In *Proc. of HLT-NAACL*, pages 1–10.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013b. Distributed representations of words and phrases and their compositionality.

George A. Miller, R.T. Beckwith, Christiane D. Fellbaum, D. Gross, and K. Miller. 1990. Introduction to WordNet: an online lexical database. *International Journal of Lexicography*, 3(4):235–244.

George A Miller, Claudia Leacock, Randee Tengi, and Ross T Bunker. 1993. A semantic concordance. In *Proceedings of the workshop on Human Language Technology*, pages 303–308. Association for Computational Linguistics.

Bonan Min, Hayley Ross, Elior Sulem, Amir Pouran Ben Veyseh, Thien Huu Nguyen, Oscar Sainz, Eneko Agirre, Ilana Heintz, and Dan Roth. 2023a. Recent advances in natural language processing via large pre-trained language models: A survey. *ACM Comput. Surv.*, 56(2).

Do June Min, Veronica Perez-Rosas, and Rada Mihalcea. 2023b. Navigating data scarcity: Pretraining for medical utterance classification. In *Proceedings of the 5th Clinical Natural Language Processing Workshop*, pages 59–68, Toronto, Canada. Association for Computational Linguistics.

Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen-tau Yih, Pang Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2023c. FActScore: Fine-grained atomic evaluation of factual precision in long form text generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12076–12100, Singapore. Association for Computational Linguistics.

Francesco Molfese, Andrei Bejgu, Simone Tedeschi, Simone Conia, and Roberto Navigli. 2024. Crocoalign: A cross-lingual, context-aware and fully-neural sentence alignment system for long texts. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2209–2220, St. Julian's, Malta. Association for Computational Linguistics.

Robert C. Moore. 2002. Fast and accurate sentence alignment of bilingual corpora. In *Proceedings of the 5th Conference of the Association for Machine Translation in the Americas: Technical Papers*, pages 135–144, Tiburon, USA. Springer.

Andrea Moro and Roberto Navigli. 2013. Integrating syntactic and semantic analysis into the

open information extraction paradigm. In *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*, IJCAI '13, page 2148–2154. AAAI Press.

Andrea Moro and Roberto Navigli. 2015. SemEval-2015 task 13: Multilingual all-words sense disambiguation and entity linking. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 288–297, Denver, Colorado. Association for Computational Linguistics.

David Mueller, Nicholas Andrews, and Mark Dredze. 2020. Sources of transfer in multilingual named entity recognition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8093–8104, Online. Association for Computational Linguistics.

Dor Muhlgay, Ori Ram, Inbal Magar, Yoav Levine, Nir Ratner, Yonatan Belinkov, Omri Abend, Kevin Leyton-Brown, Amnon Shashua, and Yoav Shoham. 2024. Generating benchmarks for factuality evaluation of language models. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 49–66, St. Julian's, Malta. Association for Computational Linguistics.

Masaaki Nagata, Katsuki Chousa, and Masaaki Nishino. 2020. A supervised word alignment method based on cross-language span prediction using multilingual BERT.

Roberto Navigli. 2009. Word sense disambiguation: A survey. *ACM computing surveys (CSUR)*, 41(2):1–69.

Roberto Navigli. 2012. A quick tour of word sense disambiguation, induction and related approaches. In *SOFSEM 2012: Theory and Practice of Computer Science: 38th Conference on Current Trends in Theory and Practice of Computer Science, Špindlerv Mlỳn, Czech Republic, January 21-27, 2012. Proceedings 38*, pages 115–129. Springer.

Roberto Navigli, Michele Bevilacqua, Simone Conia, Dario Montagnini, and Francesco Cecconi. 2021. Ten years of BabelNet: A survey. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 4559–4567. International Joint Conferences on Artificial Intelligence Organization. Survey Track.

Roberto Navigli, David Jurgens, and Daniele Vannella. 2013. SemEval-2013 task 12: Multilingual word sense disambiguation. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 222–231, Atlanta, Georgia, USA. Association for Computational Linguistics.

Steven Neale. 2018. A survey on automatically-constructed WordNets and their evaluation: Lexical and word embedding-based approaches. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Graham Neubig. 2011. The Kyoto free translation task.

NLLB team, Marta Costa-jussa, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janicec Lam, Daniel Licht, Jean Maillard, Anna Sun, Skyler Wang, Guillaume Wenzek, Al Youngblood, Bapi Akula, Loïc Barrault, Gabriel Gonzalez, Prangthip Hansanti, and Jeff Wang. 2022. No language left behind: Scaling human-centered machine translation. *ArXiv*, abs/2207.04672.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational linguistics*, 29(1):19–51.

Franz Josef Och, Christoph Tillmann, and Hermann Ney. 1999. Improved alignment models for statistical machine translation. In *1999 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings,

Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens,

Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. 2024. Gpt-4 technical report.

Riccardo Orlando, Simone Conia, Stefano Faralli, and Roberto Navigli. 2022. Universal semantic annotator: the first unified API for WSD, SRL and semantic parsing. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 2634–2641, Marseille, France. European Language Resources Association.

Riccardo Orlando, Pere-Lluís Huguet Cabot, Edoardo Barba, and Roberto Navigli. 2024. ReLiK: Retrieve and LinK, fast and accurate entity linking and relation extraction on an academic budget. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 14114–14132, Bangkok, Thailand and virtual meeting. Association for Computational Linguistics.

Daniel W. Otter, Julian R. Medina, and Jugal K. Kalita. 2021. A survey of the usages of deep learning for natural language processing. *IEEE Transactions on Neural Networks and Learning Systems*, 32(2):604–624.

Sebastian Padó and Mirella Lapata. 2009. Cross-lingual annotation projection for semantic roles. *Journal of Artificial Intelligence Research*, 36:307–340.

Artidoro Pagnoni, Vidhisha Balachandran, and Yulia Tsvetkov. 2021. Understanding factuality in abstractive summarization with frank: A benchmark for factuality metrics.

Alicia Parrish, William Huang, Omar Agha, Soo-Hwan Lee, Nikita Nangia, Alexia Warstadt, Karmanya Aggarwal, Emily Allaway, Tal Linzen, and Samuel R. Bowman. 2021. Does

putting a linguist in the loop improve NLU data collection? In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4886–4901, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Tommaso Pasini, Alessandro Raganato, and Roberto Navigli. 2021. XL-WSD: An extra-large and cross-lingual evaluation framework for word sense disambiguation. In *Proc. of AAAI*.

Maria Pelevina, Nikolay Arefiev, Chris Biemann, and Alexander Panchenko. 2016. Making sense of word embeddings. In *Proceedings of the 1st Workshop on Representation Learning for NLP*, pages 174–183, Berlin, Germany. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Robert Logan, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A. Smith. 2019. Knowledge enhanced contextual word representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 43–54, Hong Kong, China. Association for Computational Linguistics.

Sameer Pradhan, Edward Loper, Dmitriy Dligach, and Martha Palmer. 2007. SemEval-2007 task-17: English lexical sample, SRL and all words. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 87–92, Prague, Czech Republic. Association for Computational Linguistics.

Luigi Procopio, Edoardo Barba, Federico Martelli, and Roberto Navigli. 2021. Multimirror: Neural cross-lingual word alignment for multilingual word sense disambiguation. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 3915–3921.

Xiao Pu, Mingqi Gao, and Xiaojun Wan. 2023. Summarization is (almost) dead.

Ye Qi, Devendra Sachan, Matthieu Felix, Sarguna Padmanabhan, and Graham Neubig. 2018. When and why are pre-trained word embeddings useful for neural machine translation? In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 529–535, New Orleans, Louisiana. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *CoRR*, abs/1910.10683.

Alessandro Raganato, Jose Camacho-Collados, and Roberto Navigli. 2017. Word sense disambiguation: A unified evaluation framework and empirical comparison. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 99–110, Valencia, Spain. Association for Computational Linguistics.

Delip Rao, Paul McNamee, and Mark Dredze. 2013. Entity linking: Finding extracted entities in a knowledge base. *Multi-source, multilingual information extraction and summarization*, pages 93–115.

Zafaryab Rasool, Stefanus Kurniawan, Sherwin Balugo, Scott Barnett, Rajesh Vasa, Courtney Chesser, Benjamin M. Hampstead, Sylvie Belleville, Kon Mouzakis, and Alex Bahar-Fuchs. 2024. Evaluating llms on document-based qa: Exact answer selection and numerical extraction using cogtale dataset. *Natural Language Processing Journal*, 8:100083.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Maximilian Schmidt, Andrea Bartezzaghi, and Ngoc Thang Vu. 2024. Prompting-based synthetic data generation for few-shot question answering. In *Proceedings of the 2024*

Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024), pages 13168–13178, Torino, Italia. ELRA and ICCL.

Holger Schwenk, Vishrav Chaudhary, Shuo Sun, Hongyu Gong, and Francisco Guzmán. 2021. WikiMatrix: Mining 135M parallel sentences in 1620 language pairs from Wikipedia. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1351–1361, Online. Association for Computational Linguistics.

Alessandro Scirè, Karim Ghonim, and Roberto Navigli. 2024. Fenice: Factuality evaluation of summarization based on natural language inference and claim extraction.

Scirè and Bejgu, Simone Tedeschi, Karim Ghonim, Federico Martelli, and Roberto Navigli. 2024. Truth or mirage? towards end-to-end factuality evaluation with llm-oasis.

Lutfi Kerem Senel, Ihsan Utlu, Veysel Yucesoy, Aykut Koc, and Tolga Cukur. 2018. Semantic structure and interpretability of word embeddings. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(10):1769–1779.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Rico Sennrich and Martin Volk. 2011. Iterative, mt-based sentence alignment of parallel texts. In *Proceedings of the 18th Nordic conference of computational linguistics (NODALIDA 2011)*, pages 175–182.

Noam Shazeer and Mitchell Stern. 2018. Adafactor: Adaptive learning rates with sublinear memory cost. *CoRR*, abs/1804.04235.

Xuewen Shi, Heyan Huang, Ping Jian, and Yi-Kun Tang. 2021. Improving neural machine translation with sentence alignment learning. *Neurocomputing*, 420:15–26.

Samuel L. Smith, David H. P. Turban, Steven Hamblin, and Nils Y. Hammerla. 2017a. Offline bilingual word vectors, orthogonal transformations and the inverted softmax.

Samuel L. Smith, David H. P. Turban, Steven Hamblin, and Nils Y. Hammerla. 2017b. Offline bilingual word vectors, orthogonal transformations and the inverted softmax. *ArXiv*, abs/1702.03859.

Benjamin Snyder and Martha Palmer. 2004. The English all-words task. In *Proceedings of SENSEVAL-3, the Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, pages 41–43, Barcelona, Spain. Association for Computational Linguistics.

Yang Song, Xin Cai Ong, Hwee Tou Ng, and Qian Lin. 2021. Improved word sense disambiguation with enhanced sense representations. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4311–4320, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Elias Stengel-Eskin, Tzu-ray Su, Matt Post, and Benjamin Van Durme. 2019. A discriminative neural model for cross-lingual word alignment. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 910–920, Hong Kong, China. Association for Computational Linguistics.

Derek Tam, Anisha Mascarenhas, Shiyue Zhang, Sarah Kwan, Mohit Bansal, and Colin Raffel. 2022. Evaluating the factual consistency of large language models through summarization.

Liyan Tang, Tanya Goyal, Alex Fabbri, Philippe Laban, Jiacheng Xu, Semih Yavuz, Wojciech Kryscinski, Justin Rousseau, and Greg Durrett. 2023. Understanding factual errors in summarization: Errors, summarizers, datasets, error detectors. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11626–11644, Toronto, Canada. Association for Computational Linguistics.

Liyan Tang, Igor Shalyminov, Amy Wing mei Wong, Jon Burnsky, Jake W. Vincent, Yu'an Yang, Siffi Singh, Song Feng, Hwanjun Song, Hang Su, Lijia Sun, Yi Zhang, Saab Mansour, and Kathleen McKeown. 2024. Tofueval: Evaluating hallucinations of llms on topic-focused dialogue summarization.

Yixuan Tang and Yi Yang. 2024. Do we need domain-specific embedding models? an empirical investigation.

Simone Tedeschi, Valentino Maiorca, Niccolò Campolungo, Francesco Cecconi, and Roberto Navigli. 2021. WikiNEuRal: Combined neural and knowledge-based silver data creation for multilingual NER. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2521–2533, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Brian Thompson and Philipp Koehn. 2019. Vecalign: Improved sentence alignment in linear time and space. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1342–1348, Hong Kong, China. Association for Computational Linguistics.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for fact extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 809–819, New Orleans, Louisiana. Association for Computational Linguistics.

J. Tiedemann. 2007. Improved sentence alignment for movie subtitles. In *Proceedings of RANLP 07, Borovets, Bulgaria*. INCOMA Ltd. 2007/j.tiedemann/pub006.

Jörg Tiedemann. 2004. Word to word alignment strategies. In *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*, pages 212–218.

S. M Towhidul Islam Tonmoy, S M Mehedi Zaman, Vinija Jain, Anku Rani, Vipula Rawte, Aman Chadha, and Amitava Das. 2024. A comprehensive survey of hallucination mitigation techniques in large language models.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models.

Dániel Varga, Péter Halácsy, András Kornai, Viktor Nagy, László Németh, and Viktor Trón. 2007. Parallel corpora for medium density languages. *Amsterdam Studies In The Theory And History Of Linguistic Science Series 4*, 292:247.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

David Vilar, Maja Popović, and Hermann Ney. 2006. Aer: Do we need to "improve" our alignments? In *Proc. of Workshop on Spoken Language Translation*.

Stephan Vogel, Hermann Ney, and Christoph Tillmann. 1996. Hmm-based word alignment in statistical translation. In *COLING 1996 Volume 2: The 16th International Conference on Computational Linguistics*.

Hengyi Wang, Haizhou Shi, Shiwei Tan, Weiyi Qin, Wenyuan Wang, Tunyu Zhang, Akshay Nambi, Tanuja Ganu, and Hao Wang. 2024a. Multimodal needle in a haystack: Benchmarking long-context capability of multimodal large language models.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2022. Text embeddings by weakly-supervised contrastive pre-training.

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2024b. Improving text embeddings with large language models. pages 11897–11916.

Longyue Wang, Chenyang Lyu, Tianbo Ji, Zhirui Zhang, Dian Yu, Shuming Shi, and Zhaopeng Tu. 2023. Document-level machine translation with large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 16646–16661, Singapore. Association for Computational Linguistics.

Yujie Wang and Mike Izbicki. 2023. DocSplit: Simple contrastive pretraining for large document embeddings. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 14190–14196, Singapore. Association for Computational Linguistics.

Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. 2020. Unsupervised data

augmentation for consistency training. In *Advances in Neural Information Processing Systems*, volume 33, pages 6256–6268. Curran Associates, Inc.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2020. Xlnet: Generalized autoregressive pretraining for language understanding.

David Yarowsky and Grace Ngai. 2001. Inducing multilingual pos taggers and np bracketers via robust projection across aligned corpora. In *Second Meeting of the North American Chapter of the Association for Computational Linguistics*.

Wen-tau Yih, Kristina Toutanova, John C. Platt, and Christopher Meek. 2011. Learning discriminative projections for text similarity measures. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning*, pages 247–256, Portland, Oregon, USA. Association for Computational Linguistics.

Tom Young, Devamanyu Hazarika, Soujanya Poria, and Erik Cambria. 2018. Recent trends in deep learning based natural language processing. *ieee Computational intelligenCe magazine*, 13(3):55–75.

Thomas Zenkel, Joern Wuebker, and John DeNero. 2019. Adding interpretable attention to neural translation models improves word alignment. *arXiv preprint arXiv:1901.11359*.

Yuheng Zha, Yichi Yang, Ruichen Li, and Zhiting Hu. 2023. Alignscore: Evaluating factual consistency with a unified alignment function.

Biao Zhang, Barry Haddow, and Alexandra Birch. 2023a. Prompting large language model for machine translation: A case study.

Guobiao Zhang, Wenpeng Lu, Xueping Peng, Shoujin Wang, Baoshuo Kan, and Rui Yu. 2022a. Word sense disambiguation with knowledge-enhanced and local self-attention-based extractive sense comprehension. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 4061–4070, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J. Liu. 2020. Pegasus: pre-training with extracted gap-sentences for abstractive summarization. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org.

Shiyue Zhang and Mohit Bansal. 2021. Finding a balanced degree of automation for summary evaluation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6617–6632, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Tianyi Zhang, Faisal Ladhak, Esin Durmus, Percy Liang, Kathleen McKeown, and Tatsunori Hashimoto. 2023b. Benchmarking large language models for news summarization. *Transactions of the Association for Computational Linguistics*, 12:39–57.

Wenxuan Zhang, Yue Deng, Bing Liu, Sinno Pan, and Lidong Bing. 2024. Sentiment analysis in the era of large language models: A reality check. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 3881–3906, Mexico City, Mexico. Association for Computational Linguistics.

Wenzheng Zhang, Wenyue Hua, and Karl Stratos. 2022b. EntQA: Entity linking as question answering. In *International Conference on Learning Representations*.

Michał Ziemski, Marcin Junczys-Dowmunt, and Bruno Pouliquen. 2016. The United Nations parallel corpus v1.0. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3530–3534, Portorož, Slovenia. European Language Resources Association (ELRA).

Pierre Zweigenbaum, Serge Sharoff, and Reinhard Rapp. 2017. Overview of the second BUCC shared task: Spotting parallel sentences in comparable corpora. In *Proceedings of the 10th Workshop on Building and Using Comparable Corpora*, pages 60–67, Vancouver, Canada. Association for Computational Linguistics.