A Multi-sense Context-agnostic Definition Generation Model Evaluated on Multiple Languages

by

Arman Kabiri

Bachelor of Computer Software Engineering, Shahrekord University, 2017

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

Master of Computer Science

In the Graduate Academic Unit of Faculty of Computer Science

Supervisor(s):	Paul Cook, PhD, Computer Science
Examining Board:	Huajie Zhang, PhD, Computer Science, Chair
	Mike Fleming, PhD, Computer Science
	Christine Horne, PhD, French

This thesis is accepted by the
Dean of Graduate Studies

THE UNIVERSITY OF NEW BRUNSWICK

August, 2020

Abstract

Definition modeling is a recently-introduced task in natural language processing (NLP) which aims to predict and generate dictionary-style definitions for any given word. Most prior work on definition modelling has not accounted for polysemy — i.e. a linguistic phenomenon in which a word can imply multiple meanings when used in various contexts — or has done so by considering definition modelling for a target word in a given context. In contrast, in this study, we propose a context-agnostic approach to definition modelling, based on multi-sense word embeddings, that is capable of generating multiple definitions for a target word. In further contrast to most prior work, which has primarily focused on English, we evaluate our proposed approach on fifteen different datasets covering nine languages from several language families. To evaluate our approach we consider several variations of BLEU — i.e., a widely-used evaluation metric initially introduced for machine translation that is adapted to definition modeling. Our results demonstrate that our proposed multi-sense model outperforms a single-sense model on all fifteen datasets.

Dedication

To my grandmother, who is the symbol of resistance and devotion to me. I am certain she has been and she will be the most admirable woman in my life.

To my mother, who I owe my identity to. She has been the most impressive and inspiring person in my life. Maman, I LOVE you to the moon and back!

To my sister, Atoosa, who is the LOVE of my life. Azizam, I need you to know that there is not any single day when I do not miss you and do not think about you. I am eagerly looking forward to having you join me here.

And to my beloved friend, who has gracefully and kindly opened new doors to my life.

Acknowledgements

First and foremost, I wish to thank my parents and my two beloved uncles who raised me to become the person that I am today. I appreciate and acknowledge their devotion and their support during my entire life. I have tremendous gratitude for my supervisor, Professor Cook, who has devoted many hours to help me reach the goal of my studies and has gracefully provided me with his valuable knowledge and experience. Besides, as a supervisor, his patience and tact has been admirable to me. I would also like to thank my lab partners and friends, Ali and Milton, who have always been there to support and have never hesitated to share their knowledge with me. I clearly remember the times when Ali spent hours discussing my research-related problems. Last but not the least, I am grateful for my beloved friends' emotional support throughout my entire studies.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Natural language processing (NLP) is a sub-field of computer science, and more specifically artificial intelligence, which concerns enabling computers to learn, understand, and even generate natural (human) language data. The history of NLP probably gets back to the 1950s when Alan Turing published an article entitled *Computing Machinery and Intelligence* that introduced what is now called the Turing test as a benchmark for intelligence [88]. The Turing Test was proposed to evaluate the ability of computers in exhibiting intelligent behaviour equivalent to, or indistinguishable from, that of a human. In this test, a human evaluator communicates with a human and a computer designed to generate human-like responses, separately and through a text-only channel. The human evaluator knows that one of the partners in the conversations is a machine. If the human evaluator cannot reliably recognize the artificial partner in the conversations, the computer is said to pass the Turing test. In the recent years, thanks to deep learning approaches and powerful machines with a high computational ability, the developed NLP methods have advanced significantly and they have performed as well as a human in some NLP tasks [52].

Various methods have been designed and developed to solve a wide range of NLP problems. Machine translation is one of the early tasks that was proposed in the 1950s [35]. In machine translation, the aim is to develop a system which takes a sentence or a document in a language and translates it into another language while the generated text is grammatically and semantically correct. Sentiment analysis is another popular task in NLP in which a method is designed to take a sentiment-bearing document as an input and predict a score for the given input indicating the positivity or negativity of the given text. An example input to a sentiment analysis system could be *The hotel staff were so friendly and the food was great.* It is expected that the developed method for this task predicts the *POSITIVE* label for the given sentence as the writer of the review sentence has expressed positive feelings about the hotel in the review. Another interesting task in NLP is part-of-speech (POS) tagging. The aim of this task is to propose and design methods which are able to predict proper part-of-speech tags for the tokens of a given sentence. A part of speech is a category of words that have similar grammatical properties. Some of the most renowned parts-of-speech are *noun*, *pronoun*, *verb*, *adjective*, *adverb*, *preposition*, *conjunction*, and *interjection*. For instance, for the sentence *I like to visit Iran*, the expected output could be the sequence *pronoun, verb, preposition, verb, noun.*

Generally, NLP methods deal with textual data which is composed of sequences of words. Since computers basically are binary machines and are designed to process numerical data, the textual data needs to be converted to numerical data. To represent these discrete units of language data (words) in a numerical representation, word embeddings were introduced. In earlier word embedding methods, words were mapped into very high dimensional sparse vectors, later known as one-hot vectors as in each vector only one of the dimensions corresponding to the represented word

had the value of 1 and other dimensions were set to 0. These sparse vectors were not able to capture the similarities and relationships between words. To overcome this limitation, dense distributed word embeddings were introduced. In contrast to one-hot word representations, in dense word embeddings, all dimensions of the dense word embeddings have real numbers — i.e. each word is mapped into a point in this high-dimensional vector space. An interesting property of distributed word embeddings is that they are able to capture semantic and syntactic information of words. After training word embeddings for the words seen in a training corpus, the words having similar meanings are mapped close to each other in the vector space, while the words not sharing any similarity are mapped more distant from each other. Word2Vec [60] and GloVe [72] are two early neural network based word embedding methods which have been widely used among NLP researchers.

The advent of pre-trained distributed word embeddings has pushed the boundaries in NLP further and led to significant improvements in almost every NLP task such as sentiment analysis [29], machine translation [100], question answering [58], named entity recognition [41], document classification [40], etc. Word embeddings, despite their robustness and accuracy in capturing the semantics of words through dense real-valued vectors, suffer from a limitation that they cannot differentiate between different meanings of each word. That is, they conflate all of a word's senses into a single vector. For example, the word *apple* has two unrelated meanings: *a fruit* and *the name of a technology company*. The typical word embedding methods described above assign one vector representation to this word ignoring the fact that this word can imply multiple different meanings in various contexts.

Recently, researchers have considered approaches to disambiguate word senses and learn multi-sense embeddings, in which a word is represented by multiple vectors,

each corresponding to a word's sense [8, 48]. For example, a multi-sense embedding method, in contrast to a single-sense word embedding method, learns two separate vectors for the word *apple*, one for each sense (meaning). The number of vectors learned for each word can be determined from the corpus by the method or can be fixed. The superiority of these embeddings to the traditional word embeddings is shown in many NLP tasks such as natural language understanding (NLU) tasks [50] — e.g. question answering is an NLU task in which a model is trained to comprehend a given question and a reference text and answer to the asked question, accordingly —, reverse dictionaries [32] (i.e. dictionaries which allow to search for a word by its definition), and text classification [13] — i.e. an NLP task in which a document or sentence is assigned to a predefined category. Topic classification is an example of document classification in which documents are categorized based on their contents into different pre-defined topics. More recent work has considered contextualized word embeddings, such as [23], which provide a vector representation for a given word based on the context the word is used in. For example, for the word *apple* in the two sentences *Everyday, I try to eat at least one apple* and *I like Apple products as they are very high-tech and elegant*, two separate word embeddings are calculated based on the other words surrounding the target word (*apple*) in the context.

Definition modelling, recently introduced by Noraset et al. [69], is a specific type of language modelling which aims to generate dictionary-style definitions for a given word. Definition modelling can be served as an evaluation tool for word embeddings by providing a transparent interpretation of the information represented in them. For example to make sure that a word embedding method has captured the semantics of the word *river* correctly, we need to give the word embedding of this word to a trained definition modeling system and evaluate its output. If the definition generated by the system for the given word (*river* in this example) matches the real

meaning of this word (*a natural stream of water*), we can conclude that the word embedding method has been successful in capturing the semantic information of the given word. Furthermore, a potential application of definition modeling in lexicography is to keep dictionaries updated by generating definitions for newly-emerged words that are not yet recorded in dictionaries. An example of such a word is *twitter* whose conventional meaning is *to talk quickly in a high excited voice, especially about something that is not very important* [1]. However, this word has taken another different meaning which is *A popular social media*. This second meaning is not recorded yet in some of the existing dictionaries and is missing. For example it is not included in WordNet [63], a widely used computational lexicon.

The approach to definition modelling of [69] is based on a recurrent neural network (RNN) language model, which is conditioned on a word embedding for the target word to be defined, specifically pre-trained word2vec [60] embeddings. As such, this model does not account for polysemy — i.e., words can have multiple meanings depending on the context in which they are used. To address this limitation, a number of studies have proposed context-aware definition generation models [68, 28, 37, 59, 17]. In all of these approaches, the models generate a context-specific definition for the given target word using contextual information obtained from the given context. One of the main limitations of these models is that they are absolutely dependent on the appearance of the words in contexts to be able to predict sense-specific definitions for the given target word. In fact, they are not applicable in cases when word usages — i.e., contexts in which target words appear — are not available. The reason that it matters is that depending on the application, sometimes we need a model which is able to generate multiple definitions that a word can imply in various contexts, rather than generating one context-specific definition for the word used in a specific given context.

In contrast, in this work, we propose a context-agnostic multi-sense definition generation model. Given a target word, without providing its usage in a specific context, the proposed model generates multiple definitions corresponding to different senses of that word. Our proposed model is an extension of [69] that incorporates pre-trained multi-sense embeddings. As such, the definitions that are generated are based on the senses learned by the embedding model on a background corpus, and reflect the usage of words in that corpus. Under this setup — i.e., generating multiple definitions for each word corresponding to senses present in a corpus — the proposed definition generation model has the potential to generate partial dictionary entries. In order to train the proposed model, pre-trained sense vectors for a word need to be matched to reference definitions for that word. We consider two approaches to this matching based on cosine similarity between sense vectors and reference definitions.

Following [99], we evaluate our proposed model using variations of BLEU [71]. BLEU is a widely-used evaluation measure for machine translation which basically looks for overlaps between a machine-generated translation and multiple reference definitions. It is also widely used for evaluating definition generation models — a machine-generated definition for a given target word is compared against multiple reference definitions associated to the given word in a dictionary. We evaluate our model on fifteen datasets covering nine languages from several families. Our experimental results show that, for every language and dataset considered, our proposed approach outperforms the benchmark approach of [69] which does not model polysemy.

In this study, the main research question that we have aimed to answer is *Do multi-sense embeddings enable definition generation models to generate multiple sense-specific definitions for polysemous words?* The short answer to this question is *Yes.*

The results of our experiments provided in Chapter 5 demonstrate that the proposed multi-sense model utilizing multi-sense embeddings is able to generate multiple definitions for different senses of polysemous words. The second research question that we intend to answer in this study is *is the proposed multi-sense model applicable to other languages than English?* The answer to this question is *Yes* as well. We have evaluated our proposed multi-sense model on 15 datasets covering nine languages from different language families. The results of our experiments show that the proposed multi-sense model outperforms the single sense model in all 15 datasets. Regardless of which language from which language family we are working with, we can improve the definition generation models by incorporating pre-trained multi-sense word embeddings.

The contributions of this work can be clearly listed as:

1. Proposing a multi-sense context-agnostic definition generation model which incorporates pre-trained multi-sense embeddings.

2. Conducting an extensive multi-lingual evaluation of the proposed model on nine languages from different language families.

3. Extracting, pre-processing, and publishing fifteen datasets covering nine languages for the definition modeling task.

4. Publishing trained multi-sense definition generation models for future research work.

The remainder of this thesis is organized as follows. In Chapter 2, we review the background and related work. In Chapter 3, we elaborate on our proposed multi-sense definition generation model. In Chapter 4, we describe the setup we use to train and evaluate our models. Chapter 5 presents the quantitative and qualitative

results of our experiments on multiple languages. Finally, in Chapter 6, we briefly conclude our work.

Chapter 2

Related Work

In this section, first we present a summary of the huge amount of work done on word vector representations including traditional word embeddings, multi-sense embeddings, and recently-introduced contextualized word embeddings. A brief summary of document representation methods is also presented. Then, we review the prior work on definition modeling.

2.1 Word Vector Representations

Since the early works on natural language processing, researchers have been exploring ways to map discrete natural language units like words into numerical representations. In this section, we briefly describe the traditional count-based word representations, distributed neural network based word embeddings, fine-grained sense embeddings, and recently introduced contextualized embeddings.

2.1.1 Word Embeddings

In one of the early works on numerical word representations [31], a simple way to map discrete natural language units (i.e. words) to numerical vectors was proposed. In the proposed method, the size of the vectors is set to be equal to the vocabulary

size. Therefore, each word in the vocabulary takes an index. Then, to represent a word with the vocabulary index k, a vector is generated which stores 1 in its kth element and 0 in other elements. This method is called one-hot vector representation. To represent sentences and documents using this method, we can perform an element-wise addition on the represented vectors of its constituent words (Fig 2.1). This document representation is known as bag-of-words (BOW) model which has been used for many years in several NLP tasks [56, 12, 24].

Figure 2.1: Vector representation of a sample sentence using BOW model.

This simple approach to represent natural language objects suffers from three limitations. First, the vectors obtained using this model are too sparse which raises efficiency concerns — i.e. to represent a single word, we have to store a very high-dimensional vector. The second limitation is the inability of this method to keep information about the order of the words in a document or sentence. This is considered as a serious issue as word order plays a significant role in the meaning of the sentences. For example, the sentences *You killed the lion* and *The lion killed you* both are represented with the same vector as their constituent words are the same, while they imply very different meanings. Besides, this method is not capable of representing semantic similarity and relatedness between words. That is, the relatedness of the words belonging to the same category like *dog*, *cat*, and *mouse* is not detectable from their vector representations.

Later on, to address the limitations of the described sparse word representations,

distributed neural network based word representation methods were proposed which are known as word embeddings [61, 72]. In these methods, to each word in the vocabulary, a low-dimensional vector (i.e. vectors with typically several hundred dimensions) is assigned. There are different ways of initializing these vectors like zero initialization or random initialization using different distributions (e.g., normal or uniform distribution). In [45], the performance of different vector initializations for word embeddings is compared with respect to various NLP tasks. After the word embeddings are initialized, they are tuned using a neural network based model trained on a huge corpus of text. The main idea behind learning vector representations for words from a huge corpus of text is the fact that words can be defined in contexts by their co-occurring words. After training, the word embeddings capture semantic and syntactic information about the words in the vocabulary. For example, the words belonging to a same category are mapped close to each other in the vector space (in the upper part of Fig. 2.2 the words *dog* and *puppy* are close to each other while far from the word *houses*). This is unlike one-hot vectors, where no meaningful relation is maintained between words mapped in the vector space, and the distance between the words mapped in the vector space does not reflect any information. Moreover, an interesting property of word embeddings is that mathematical operations can be applied on them in a meaningful way. More specifically, words maintain meaningful distance from other words. For example, as shown in the lower part of Fig. 2.2, by calculating $king - man + woman$ we get *queen*. This observation demonstrates that word embeddings have properly learned the semantic fact that *king* is to *queen* as *man* is to *woman*.

Another good illustration of how words are mapped into the high-dimensional vector space is shown in Fig 2.3. As can be seen, the words with related meanings are mapped close to each other, like the words for different types of animals or types of

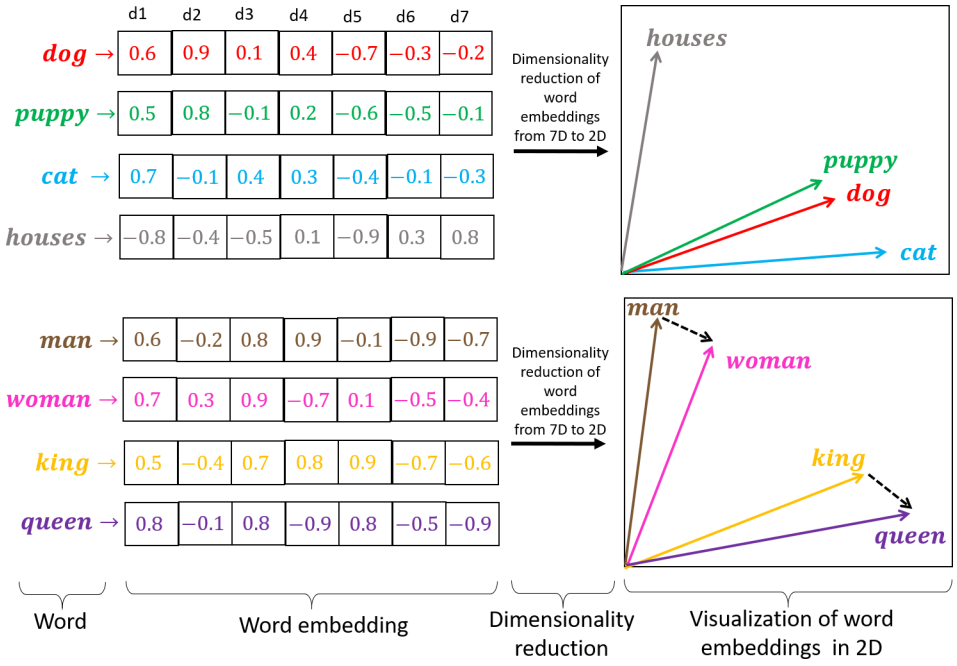

Figure 2.2: An example of the semantics captured by a word embedding method [80].

flowers.

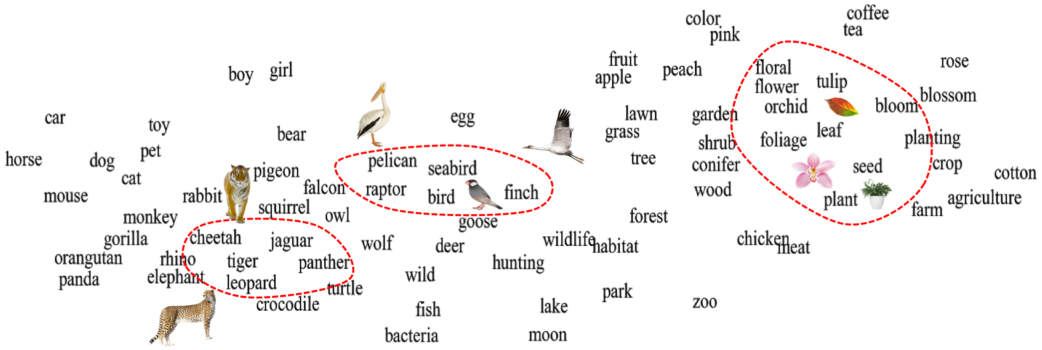

Figure 2.3: 2D illustration of the semantics captured by word embeddings [14].

Although many works have already been conducted on neural network based word embeddings whose architecture is based on optimizing a certain objective [10, 20, 87], neural network based word embeddings became widely used with the introduction of the word2vec model proposed by Mikolov et al. [60]. One of the most significant improvements of the word2vec model to the previous methods could be its amazingly high efficiency. Word2vec is a simple but efficient model which is presented through

two different but related models: continuous Bag-Of-Words (CBOW) and skip-gram. The CBOW model is an extension to the BOW model which aims to predict the current word using its surrounding words appearing in a fixed-size context window by minimizing the following loss function:

$$E = -log(p(w_t|W_t)) \tag{2.1}$$

where w_t is the target word and $W_t = w_{t-n}, ..., w_t, ..., w_{t+n}$ is the sequence of words in the context window. The general simplified architecture of the CBOW model is shown in Fig. 2.4.

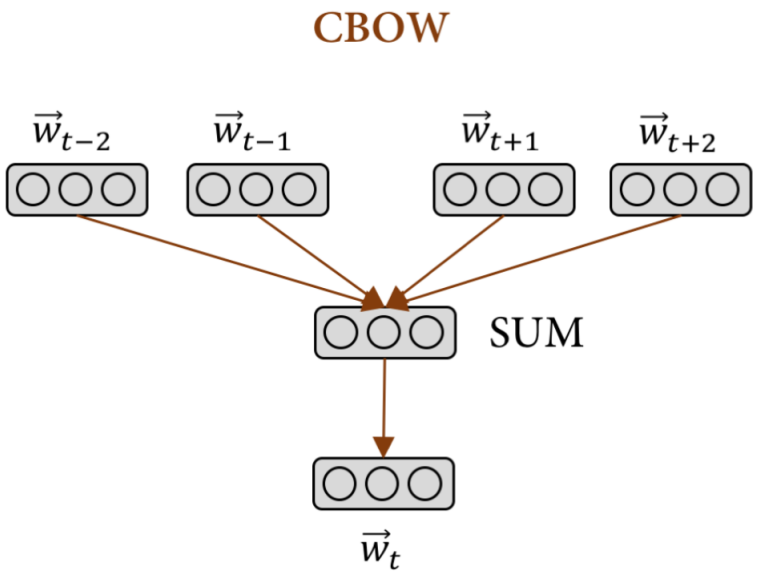

Figure 2.4: Learning architecture of the CBOW model of word2vec for a window size of 5 [60].

The skip-gram model is similar to the CBOW model; however, this model aims to predict the context words appearing in the context window given the target word. This becomes slightly tricky since we have multiple words in the context of the target word. In the skip-gram model, the (target, context words) pairs are broken down to (target, context word) pairs such that the target word forms separate pairs with each of the context words. The skip-gram model is considered as a classification problem,

where the model is trained to predict whether the given pairs of words can occur in the same context or not. The positive training samples given to the model are in the form of (x, y), where x is the pair of (target, context) and y is 1. To train the model, we also need to provide it negative examples. To build negative examples, we randomly pick a word from the vocabulary and pair it with the current target word. So x in the training samples (x, y) is the pair of (target, random word) and y is set to 0. The simplified architecture of the Skip-gram model is shown in Fig. 2.5.

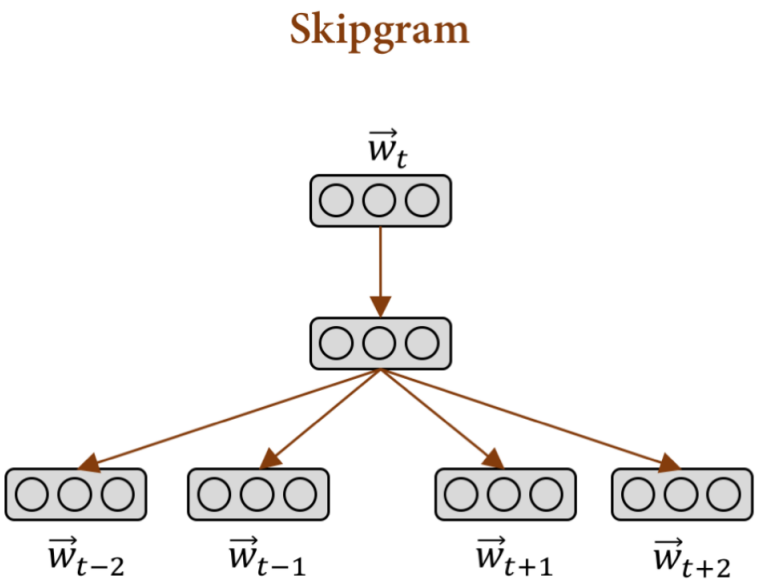

Figure 2.5: Learning architecture of the Skip-gram model of word2vec for a window size of 5 [60].

Another successful word embedding method is GloVe [72] which, unlike Word2vec, does not rely just on local context information of words, but incorporates global statistics (word co-occurrence statistics) to obtain word vectors. To obtain the global statistics, we need to form a $V * V$ co-occurrence matrix X, where the element X_{ij} corresponds to the number of times that words with the indices i and j have co-occurred. Note that V refers to the size of vocabulary. The co-occurrence matrix X for an example sentence *the cat sat on the mat* is shown in Fig. 2.6. Note that

rows and columns corresponding to other words of the vocabulary are dropped in the shown co-occurrence matrix.

	the	cat	sat	on	mat
the	0	1	0	1	1
cat	1	0	1	0	0
sat	0	1	0	1	0
on	1	0	1	0	0
mat	1	0	0	0	0

Figure 2.6: An example of co-occurrence matrix used in [72].

The main idea behind using global statistics in GloVe is to address a limitation of word2vec [60] in taking word co-occurrence in the entire corpus into consideration. For example, the words *the* and *cat* may get used together often, but word2vec cannot realize if this is because *the* is a common word or if this is because *the* has a strong linkage with *cat*. To calculate the probability of seeing words i and j with each other, we need to divide the number of times words i and j are seen with each other (X_{ij}) by the number of times that word i has appeared in the corpus ($\sum_{k=0}^{V} X_{ik}$).

2.1.2 Multi-sense Embeddings

One of the major limitations of the word embeddings described in Section 2.1.1 is that they maintain only one vector representation for each word type, meaning that polysemy — i.e. a linguistic phenomenon in which a word type can imply different meanings depending on the context it is used in — is ignored. This limitation

causes two problems. First, less frequent meanings of words are not captured, and vectors tend to represent the most frequent meaning of each word. For instance, the word *bank* can imply two different meanings in different contexts: a financial institution, or a land alongside a river. Depending on the training corpus the embeddings are trained on, one of the mentioned meanings may be seen more frequently and the other less frequent one will not be represented in the resulting embedding. Second, following this setting, semantically-unrelated words may get pulled towards each other in the vector space if they are similar to different senses of a third word, which is not desirable [14]. For example, the words *rat* and *keyboard* which are two semantically-unrelated words that may be pulled towards each other because of their similarity to the different unrelated senses of the word *mouse*, i.e., rodent and computer input device (Fig. 2.7). These two issues could affect the performance of an NLP system using these word embeddings to represent words.

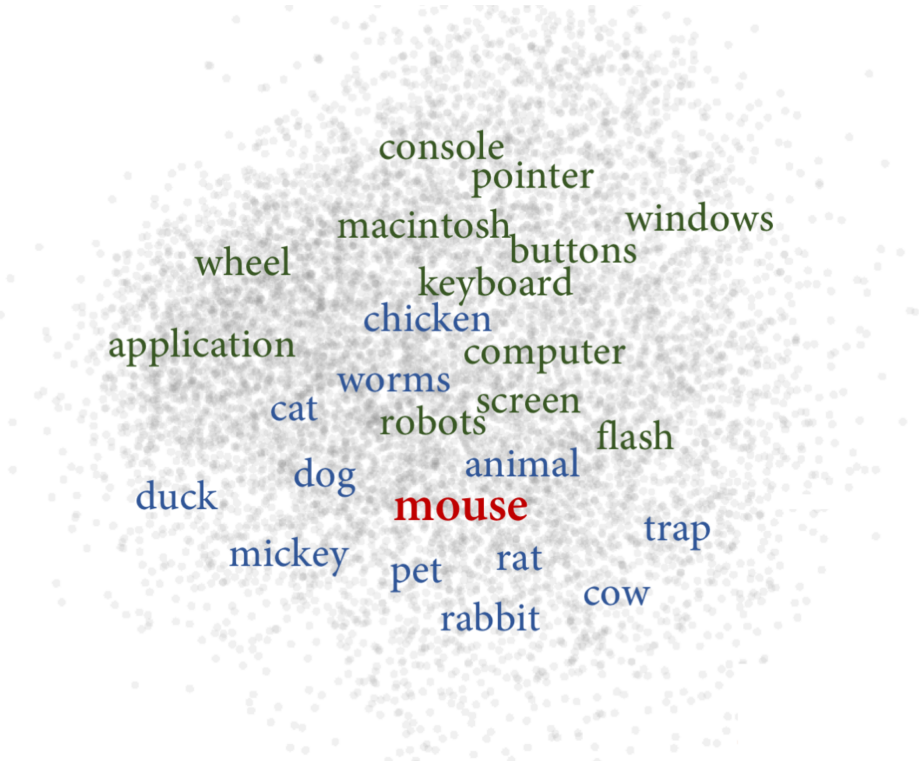

Figure 2.7: An illustration of the semantic deficiency of word embeddings [14].

To address this major limitation of word embeddings, a new research direction has attracted NLP researchers' attention over the past years, which tries to explore the ways to represent individual word senses [57, 25, 50, 66, 86]. These new techniques embedding individual word senses into distributed dense vectors are called multi-sense embedding methods. Studies on multi-sense embeddings can be divided into two categories: unsupervised methods and knowledge-based methods.

In unsupervised approaches, a model is trained over a huge corpus of raw unlabeled natural language text, e.g. a Wikipedia dump[1] or Google News,[2] to first cluster the instances of each word type into different semantic categories and then learn vector representations for different senses of each word type based on the context the target words are used in. These methods can be parametric or non-parametric, which means they need the user to specify the number of senses to learn for each word type or they can learn it during the training process, respectively. In one of the early works, Reisinger and Mooney [79] introduce a method which constructs multiple high-dimensional sparse vectors embedding multiple senses of each word. Afterwards, Huang et al. [34] utilize recurrent neural networks for using contexts to construct multiple dense sense embeddings for words. Neelakantan et al. [66] propose the first multi-sense extension to the Skip-gram model [60] which jointly trains multi-sense embeddings alongside global embeddings using the contexts in which words appear. Unlike the previous works, this method is much more efficient and as claimed, could be trained on a corpus of nearly one billion tokens in just six hours. Bartunov et al. [8] propose an extension to the skip-gram model [60] which takes the ambiguity of the words into account. Their proposed model (Adaptive Skip-gram) is a non-parametric — i.e. does not need the user to specify the number of senses to learn for each type — Bayesian extension of Skip-gram which, unlike most other

[1] `https://dumps.wikimedia.org`
[2] `https://news.google.com/`

multi-sense vector representations, automatically learns the required number of vectors to represent each word.

Most works on multi-sense embeddings only focus on sense-specific representation learning. However, incorporating multi-sense embeddings into NLP tasks requires more steps after learning sense-specific representations. Li and Jurafsky [50] conduct a study on multi-sense embeddings considering two more steps after learning sense-specific representations: sense induction and representation acquisition for phrases or sentences. They build a model on top of [34, 66] which has the property that a word is associated to a new sense vector only when a context the word is seen in proves that the previously associated sense vectors of the target word are not sufficiently relevant to the current context. To design this model, they are inspired by Chinese Restaurant Processes (CRP) [30] which says the current person (word in this application) could either sit at one of the existing tables (each belonging to one of the existing senses, in this application) or choose a new table (a new sense, in this application). To evaluate their multi-sense embedding method, they apply their model on part-of-speech tagging, named entity recognition, sentiment analysis, semantic relation identification and semantic relatedness. They demonstrate that their proposed model outperforms the previous work on most of the mentioned extrinsic NLP tasks. Recently, Lee and Chen [48] propose a fully unsupervised approach for constructing fine-grained sense embeddings. The proposed system, MUSE, exploits reinforcement learning for implementing the two suggested key factors for a multi-sense word representation system: a sense selection and a sense representation mechanism. Their proposed model achieves state-of-the-art results at the time of writing of their paper; however, it is parametric and the number of senses to learn for each word type is fixed and given by the user.

Among works on multi-sense embeddings, some of the studies have focused on the construction of embeddings for relatively coarse-grained senses which could be thought of as semantic cluster centroids. For example, in [67], Nguyen et al. propose a mixture model which learns embeddings for a fixed number of topics. The construction of the embeddings for each word is performed using a weighted combination of the learned coarse-grained senses. Following their work, Arora et al. [4] show how various senses of each word reside in a linear superposition within the vector space of standard word embedding models such as Word2Vec [60]. Their method is built upon the random walk on discourses model [5] which uses sparse coding on word embeddings to recover proper representations for various downstream NLP tasks such as word sense induction and word similarity in context. Similarly, in a recent work [13], the authors propose a distributional semantic model (DSM) learning multiple dense distributional vector representations for each word type based on different topics. For each topic, first, a separate DSM is trained. Then, each of the separate trained DSMs is aligned to a common vector space. To map the topic-based DSMs into a shared vector space, they propose an unsupervised mapping approach which is inspired by the hypothesis that words maintaining their distances in different topic-based vector spaces constitute strong semantic anchors which are used to define the mappings between them. The proposed aligned topic-based representations outperform the prior work for the task of contextual word similarity. Contextual word similarity is a method to estimate the semantic similarity between a pair of words provided in sentential context. In this work, the standard evaluation Stanford Contextual Word Similarity (SCWS) dataset [34] is used for evaluation.

The second category of multi-sense embedding methods are knowledge-based methods which take advantage of information extracted from an external sense inventory, e.g., WordNet [63], to learn sense-specific vector representations for words. A sense

inventory is a lexical resource which lists, for each word, different meanings it can imply in different contexts. In one of the works attempting to extract and incorporate knowledge from a sense inventory to sense-specific representation learning, the authors propose a technique to apply on previously existing word embedding methods [39]. The proposed approach applies a post-processing step using graph smoothing to de-conflate different senses of a word from the word vector. This proposed approach is applicable to any vector space model. The results of their experiments demonstrate that their proposed method is able to effectively capture information from both the ontology and distributional statistics. The results also show that their multi-sense embedding method outperforms previous works in most cases. Following their work, in another work exploiting sense inventories to train sense representations [74], the authors propose a method which takes pre-trained word embeddings, e.g. word2vec [60], and tries to de-conflate a given word representation into its constituent sense representations by utilizing semantic knowledge from WordNet. This employment of an external sense inventory benefits the model from two perspectives. First, the multi-sense embeddings method can be non-parametric, meaning that neither a user should fix the number of sense representations to learn for each word, nor does the model need to predict it from the corpus. Second, the learned sense representations are linked to manually-checked senses which decreases the emergence of inaccurate or duplicated senses for a word. Despite the advantages noted for the methods utilizing an external lexical resource, these methods suffer from some limitations. The dependence of these methods to an external lexical resource limits them to learning representations only for the words and word senses present in the resource. Obviously, this category of methods are not applicable in our work since we intend to train a definition generation model which is capable of generating sense-specific definitions for unseen words or new senses of previously-seen words. Benefiting from the semantic network of WordNet, this method outperforms

the previous methods in terms of Spearman [64] and Pearson [9] correlations on four standard word similarity benchmarks. Spearman and Pearson correlation are statistical methods which are commonly used to measure the quality of trained word embeddings against word similarity benchmarks like RG-65 [81], YP-130 [92], etc. Word similarity task is a famous computationally efficient benchmark for evaluating the quality of word vectors. This task is designed to find the correlation between human assigned semantic similarity score (between words) and the similarity of the corresponding word vectors.

2.1.3 Contextualized Word Embeddings

All embedding methods that we have described so far learn vector representations for word types. The traditional word embeddings learn one representation for each type, while the multi-sense word embedding methods assign multiple representations to each type. The aim of the contextualized word embeddings is to take the lexical ambiguity of natural language, specifically polysemy, into account. Contextualized word embedding methods assign vector representations to tokens — instead of word types in context-agnostic word embedding methods — based on the context the token is used in. To obtain context-dependent vector representations for words in contextualized word embedding methods, the main idea is to get the representations for tokens from the hidden states of a trained language model. Language models are probabilistic models computing the probability distribution of a word in a sequence given the previous words appearing in the sequence.

ELMo, standing for the Embeddings from Language Models, is one of the early successful works on contextualized word embeddings [73] which achieves extremely impressive results on multiple NLP benchmark tasks and pushes the state-of-the-art. ELMo is an RNN-based language model using a stacked bi-directional long

short-term memory (LSTM) network in its core. ELMo is trained by reading millions of sentences of a huge corpus both forward and backward. In fact, it has two language models, each reading the sentences in opposite directions. ELMo uses residual connections between the LSTM layers which allows gradients to flow through a network directly to the embedding layer. After the language model is trained, the contextualized representation of the kth word in a given context is computed with a concatenation and a weighted sum as shown in Equation 2.2 and Fig. 2.8

$$ELMo_k = \gamma^{task} \sum_{j=0}^{L} s_j^{task} h_{kj}^{LM} \tag{2.2}$$

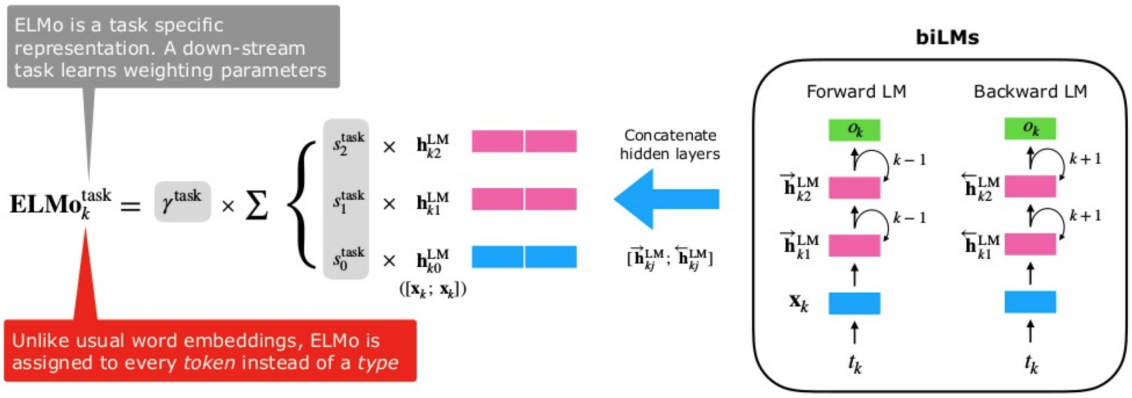

Figure 2.8: An illustration of calculation of ELMo embeddings [91].

where γ^{task} is a scalar parameter which scales the entire ELMO vector, s_j^{task} are softmax-normalized weights, and the indices k and j correspond to the index of the word in the given context and the index of the layer which the hidden state is being extracted from. Although ELMo embeddings could totally replace the context-agnostic word embeddings in practice, it is recommended by the authors to concatenate ELMo embeddings with context-independent embeddings such as from word2vec. This way, ELMo embeddings can be utilized for almost every NLP task without changing the architecture of the evaluated model (Fig. 2.9).

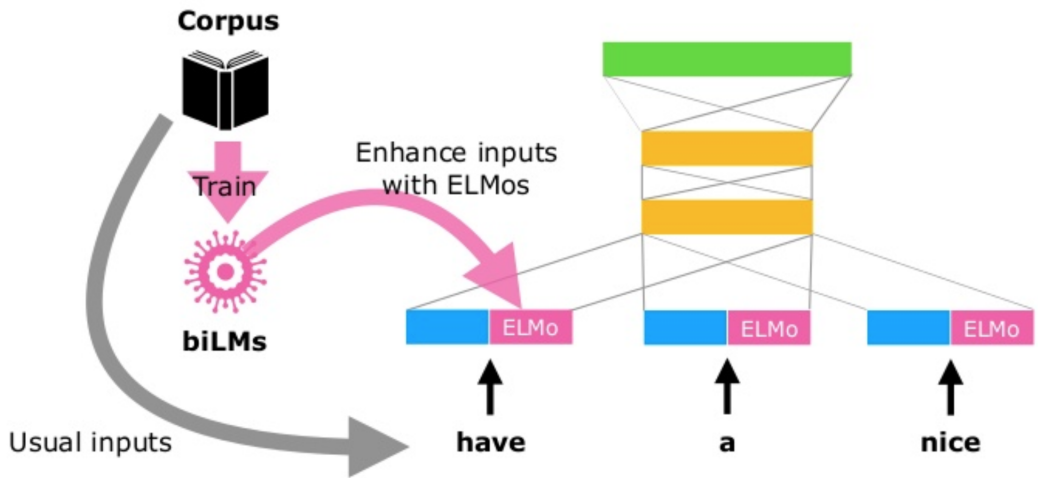

Figure 2.9: An illustration of the utilization of ELMo embeddings in any NLP model [91].

Before we explain other contextualized word embedding methods, we need to briefly describe a new neural network architecture called transformers that is used in these new contextualized word embedding methods.

A transformer is a revolutionary novel neural network architecture for language understanding which has significant advantages over the conventional sequential models (RNN, LSTM, GRU, etc) [89]. Specifically, transformers are more effective in modeling long term dependencies between tokens in a sequence. Removing the sequential dependency on previous tokens, transformer architecture is more efficient in training the language models in general. Basically, a transformer is made of an encoder and decoder utilizing attention mechanisms to pass a more comprehensive knowledge of the whole input sequence to the decoder at once rather than sequentially in sequential models such as LSTM (Fig. 2.10). More detailed explanations of transformers architecture can be found in the original paper [89].

Although OpenAI's GPT [76] was the first work to create a transformer based language model with fine tuning, BERT is the first bidirectional transformer-based

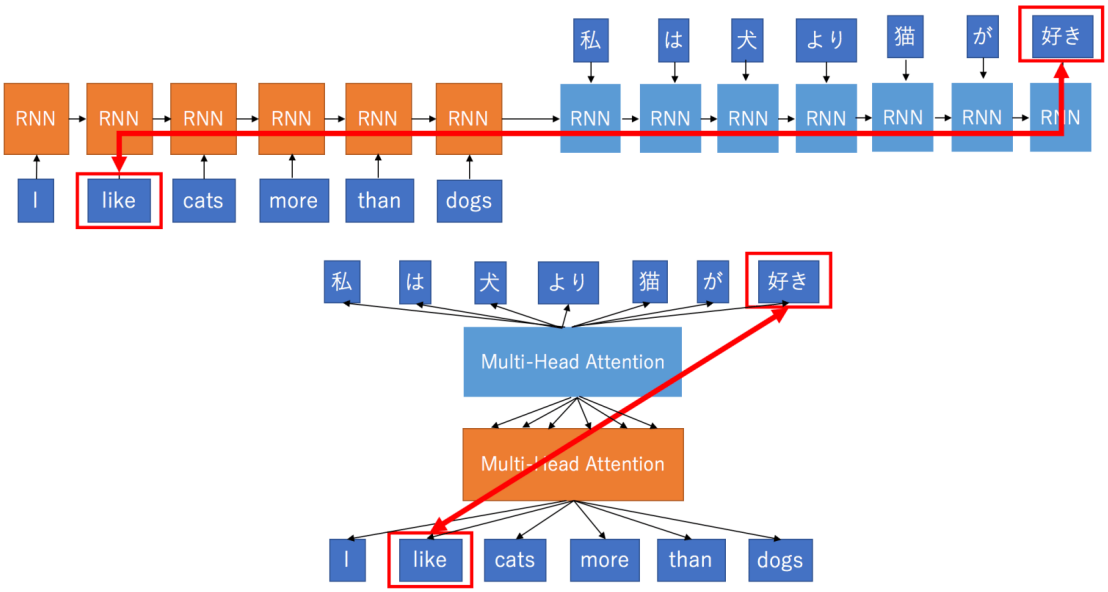

Figure 2.10: Comparison of a sequential model with a transformer in terms of passing the encoded knowledge of the input sequence to the decoder [42].

language model which only uses the encoder part of the transformers rather than the decoder [23] — in causal (traditional) language models (CLMs), as opposed to BERT which is a masked language model (MLM), each token is predicted by a decoder conditioned on the previous tokens. The main innovation of BERT is to apply the bi-directional training of transformers to language modeling. Some argue that BERT is not considered as a bi-directional language model — in contrast to previous efforts in other RNN-based language models which combine left-to-right and right-to-left training — but a non-directional language model as it reads the whole sequence at once [3]. Token prediction in BERT, as opposed to an RNN-based language model in which a token is predicted given previously seen tokens in the sequence, is implemented using a novel technique named Masked Language Modeling (MLM). In this technique, at each iteration, 15% of words are masked and are predicted using their relative position in the sequence and the other unmasked words. From a very high-level perspective, BERT's architecture is depicted in Fig. 2.11. As can be seen in Fig. 2.11, 12 layers of transformer encoders are stacked on top of each other.

Each of these encoder blocks encapsulates a more complicated model architecture which we do not intend to explain in detail as BERT, normally, is a language model, and in this thesis, we only need to understand the contextualized word embeddings extraction side of it.

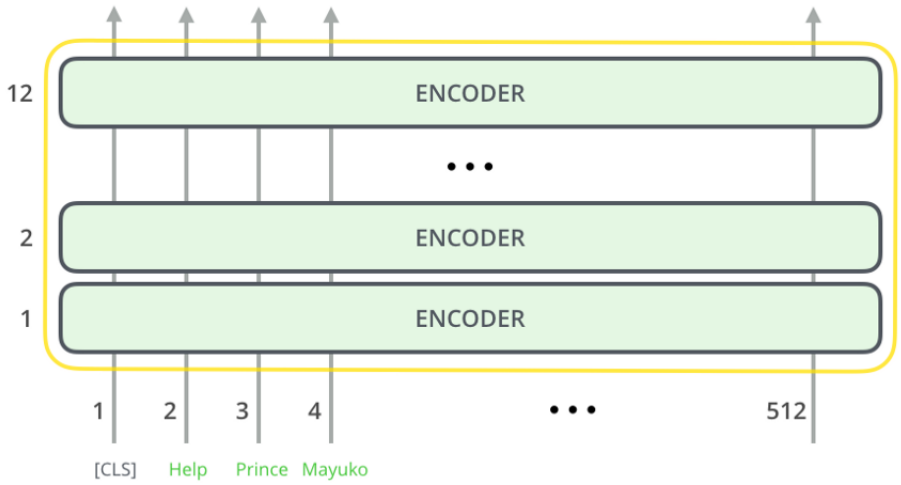

Figure 2.11: A very high-level illustration of the architecture of BERT-Base [75].

As can be seen in 2.11, words of a sequence are given to the model. The way the words of the input sequence are formed is illustrated in Fig. 2.12. As shown in Fig. 2.12, the input embeddings are obtained by summing three embeddings: token embeddings, sentence embeddings, and position embeddings. Token embeddings are the indices of the word types in the vocabulary. Sentence embeddings are just boolean embeddings indicating the sentence the given token belongs to — i.e. vectors filled with only two numbers indicating whether the corresponding token belongs to sentence A or B of the given input (shown in Fig. 2.12). Lastly, transformer positional embeddings are used to indicate the relative position of each token in the given sequence. Furthermore, in Fig. 2.12, some special tokens are seen such as [CLS] and [SEP]. [CLS] is placed at the beginning of each input which is used for classification tasks, while [SEP] is used to indicate the boundary of the two given sentences — in the cases where input only contains one sentence, [SEP] is placed at the end of the

input. Finally, to extract the BERT contextual embedding of each token, we can use the output vectors of each of the 12 BERT encoder blocks. However, based on the experiments conducted by the authors, it is suggested to concatenate the output embeddings of the last 4 encoders.

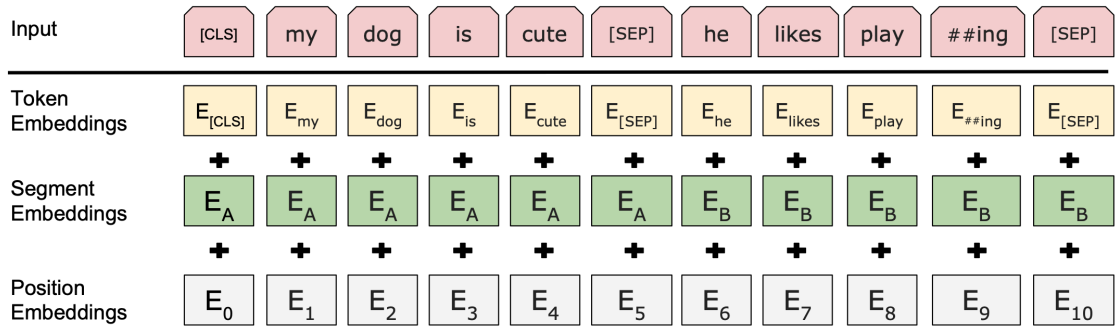

Figure 2.12: BERT input representation. The input embeddings are the sum of the token embeddings, the segmentation embeddings, and the position embeddings [23].

The transformer architecture [89], following the significant achievements of BERT [23] as the first language model using this novel architecture, caught NLP researchers' attention from all over the world. Following BERT, many other contextualized language models utilizing transformers have been introduced each presenting some advantages over other models, such as GPT [76], GPT-2 [77], XLNet [94], DistilBERT [82], and BART [49].

2.2 Document Representations

Word embeddings discussed in Section 2.1.1, mapping discrete words into semantic high-dimensional vector spaces, play a significant role in developing models which understand natural language text and have turned into a major component in almost every imaginable NLP task. After the word embeddings got popularized by Mikolov et al. presenting word2vec [60], a research direction toward document representation

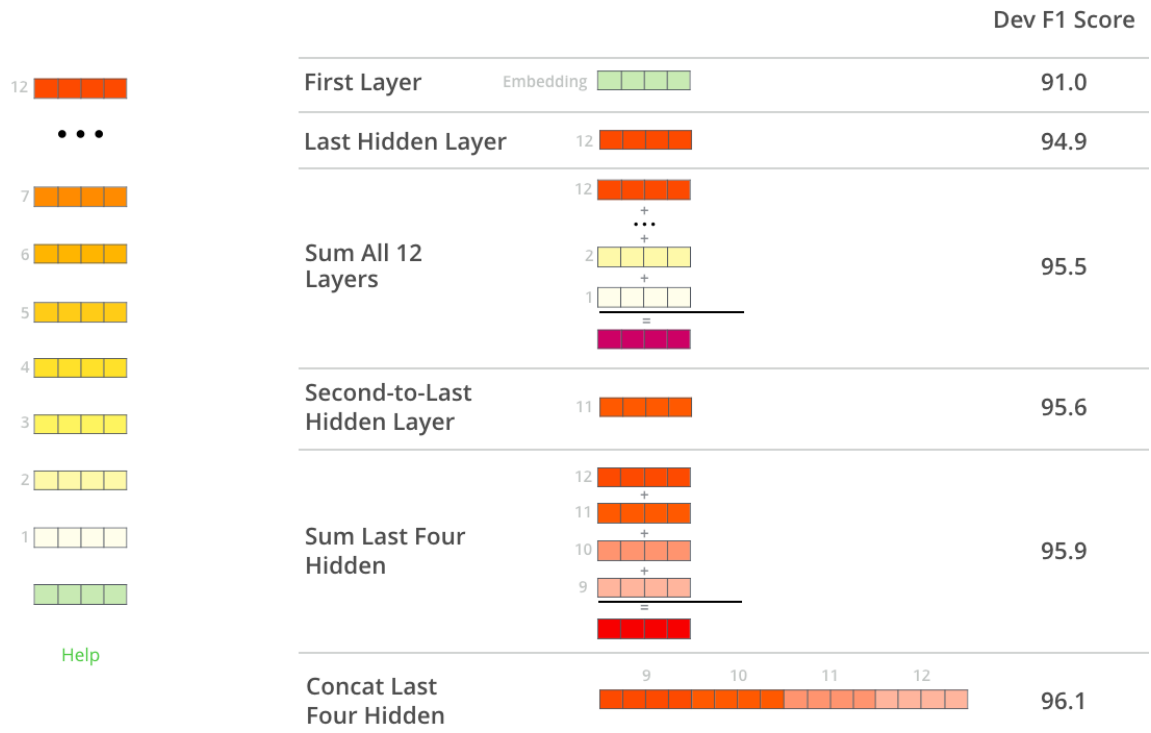

Figure 2.13: Comparison of different strategies for extracting contextualized embedding for the word *help* in context in a named-entity recognition (NER) task in terms of F1 Score [75].

resurged (the term *document* here refers to any sequence of words). In this line of research, the aim is to propose methods to capture the semantics of larger units of text, like sentences and documents, in high-dimensional vectors. Document representation methods are needed in NLP tasks which require understanding of sentences and documents. We try to provide a very brief discussion on these methods. Furthermore, in this section, we have a look into dictionary definition embedding methods in which the definitions of words are encoded to give a richer representation of the defined word.

To develop document representation methods, much work has been done which can be divided into two categories: unsupervised and supervised methods.

2.2.1 Unsupervised Methods

In unsupervised methods, no labeled data is provided and the vector representations are learned through leveraging the distributional hypothesis — i.e. words appearing in similar contexts tend to have similar meanings. The word2vec model [60] was probably the most successful method implementing this idea in a practical level. Most of the unsupervised methods for learning document representation discussed in this section are inspired by the word embedding methods, chiefly word2vec.

To represent a document, one of the most trivial approaches is to take the average of the vector representations of the constituent words of the document, which is known as continuous bag of words (CBOW) [60]. Since the CBOW model is not optimized for the task of sentence representation, it is likely to be suboptimal. To adapt CBOW model to sentence representation, Siamese CBOW is proposed [43]. Siamese CBOW tries to learn the word embeddings directly for the purpose of being

averaged. That is, the proposed model learns word embeddings by predicting context sentences using the middle sentence. The overall architecture of Siamese CBOW is shown in Fig. 2.14.

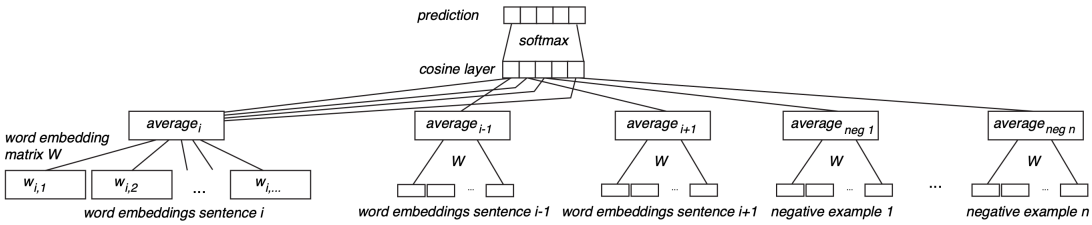

Figure 2.14: An illustration of the overall architecture of Siamese CBOW [43].

The major limitation of CBOW and Siamese CBOW is that they both ignore the ordering of the words in a document. For example, the vector representation that they learn for the two sentences *A cat ate my cute snake* and *A snake ate my cute cat* is the same, while they imply two different meanings. In [62], the authors propose to learn representations for n-grams, specifically bi-grams and tri-grams, like *brown cat* and *a brown cat* in *I have a brown cat* as individual tokens, as well. This trick addresses the mentioned limitation only to some extent and keeps the sequential information for only short dependencies. A model called Sent2Vec [70] was then proposed to combine both Siamese CBOW and the n-gram representation learning of the original CBOW. This way, they were able to optimize the learning of the word (and n-grams) embeddings for the purpose of obtaining document vectors. In another work, Socher et al. propose to use matrix-vector operations to combine the word vectors in an order given by a parse tree of a given sentence [83]. This method, as it relies on parsers, is shown to be only applicable for sentences and not paragraphs or documents. Doc2vec [47] could probably be mentioned as the first work attempting to generalize word2vec to learning representations for sequences of words in a paragraph. The authors proposed two different variations of Doc2vec: Distributed Memory (DM) and Distributed Bag of Words (DBOW). The training phase of the DM variation of Doc2vec is similar to that of CBOW [60] which is predicting a word

by its context. However, here, the context words are the preceding words, not the surrounding words. Besides, in this work, for each paragraph, a vector representation is learned to capture the topic of the paragraph. This paragraph vector is also used when predicting a word in the paragraph (Fig. 2.15). Note that the term *paragraph* here just means any sequence of words and it does not necessarily need to be a paragraph. The second variation of Doc2vec, DBOW, is an extension of the skip-gram model for document representation. In DBOW, each single context word in the context of a word in a paragraph is predicted given the paragraph vector. Note that in DBOW, word vectors are not jointly learned with paragraph vectors.

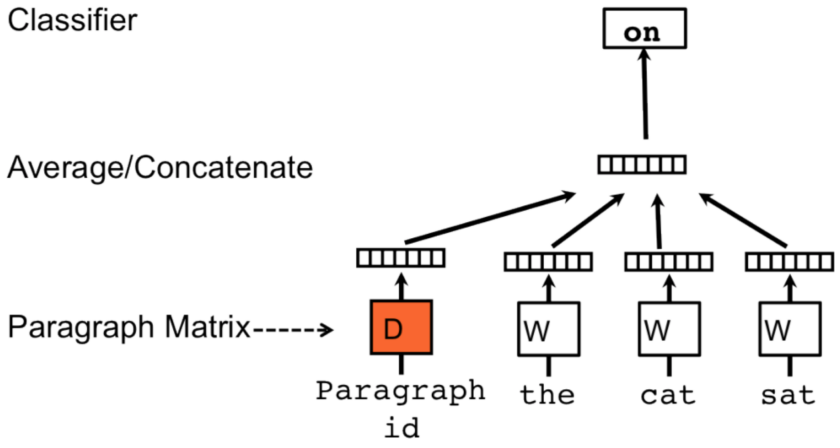

Figure 2.15: A simple illustration of the training phase of the Doc2vec-DM model [47].

As mentioned in Section 2.1.3, transformers are a new neural network architecture for modeling the sequential information of natural language text utilizing the attention mechanism [89]. Various models for natural language understanding and natural language generation have been proposed using transformers such as BERT [23] and GPT (1 and 2) [76, 77]. These models are capable of learning rich contextualized embeddings for words appearing in contexts. In addition, they can also produce representations for documents and any other given sequence of words. For example,

in BERT, there is a special token (CLS) which is used in classification tasks for capturing the entire word sequence in a fixed-length vector. However, these embeddings still turn out not be such rich document representations for NLP tasks other than classification [78]. Sentence-BERT (SBERT) is then proposed to adapt BERT to producing richer document representations [78]. In SBERT, a siamese network architecture is employed to derive semantically meaningful sentence embeddings which can be compared using cosine similarity. Siamese network is a neural network architecture which is used in one- or few- shot learning tasks where we do not have enough training data for each sample.

2.2.2 Supervised Methods

The unsupervised methods described in the previous section try to learn embeddings for documents and words without using external labels. In fact, their objective functions are designed such that they utilize the labels that are freely available within the data to learn the embeddings capturing semantic and syntactic features of the words and documents. In contrast, in the supervised approach, external data is available which is utilized by the objective function and training algorithm to learn rich word and document representations.

A notable attempt to learn document representations from labeled data is to utilize parallel corpora which is widely used for training machine translation systems. Cho et al. [19] apply an auto-encoder to explicitly learn sentence and phrase embeddings from *Europarl* [46], a parallel corpus of sentences for machine translation. Another noticeable attempt to learn document representations in a supervised manner is presented in [90]. In this work, the proposed model learns word embeddings and document representations by minimizing cosine similarity between pairs of para-

phrases extracted from the PPDB paraphrase dataset.[3]

A number of other studies have been conducted so far utilizing other sources of labeled data to learn representations for documents, such as natural language inference labeled data in [21] and question answering labeled data in [22]. However, among these works, there is a category of studies that are more relevant to the thesis, in which word–definition pairs extracted from dictionaries are utilized as stand-alone or auxiliary sources of information for training word embeddings and document representations.

Hill et al. [33] design a language model to learn vector representations for sentences and phrases. Their model is trained by predicting the words given their definitions, which is a sequence-to-word model. In this work, they have tried to propose an approach for learning useful representations of phrases. They also evaluate the quality of the learned representations using two extrinsic tasks: reverse dictionaries and general-knowledge crossword question answerers. In this study, they use two different architectures for mapping the definitions to the defined words: a bag-of-words (BOW) model, which discards the information about the order of the words in a phrase, and an RNN model. The most striking observation from the reported results is that the BOW model outperforms the RNN-based model in the reverse dictionary evaluation. The training data used in this study is also composed of five dictionaries augmented with entries from Wikipedia. In another study, Bosc and Vincent [11] propose an auto-encoder which is trained on WordNet word–definition pairs and tries to reconstruct the definitions. The embeddings extracted from this model are shown to perform better than most of the previous works in capturing semantic similarities.

[3] `http://paraphrase.org/`

2.3 Definition Modeling

Definition Modeling is a new task in natural language processing (NLP) which was first introduced in 2017 by Noraset et al. [69]. The main aim of definition modeling is to generate dictionary-style definitions for any given word. Prior work on definition modeling can be divided into two categories based on the approach they take to generate definitions which can be context-agnostic or context-aware.

2.3.1 Context-agnostic methods

In context-agnostic approaches, to produce a definition for a target word, the definition generation model is conditioned only on the vector representation of the target word, regardless of the context the word appears in. This approach may ignore polysemy and only produce definitions corresponding to the most frequent meaning of the target word captured by the word embedding methods.

The first work on definition modeling [69] proposes a context-agnostic model to learn a definition generation model for English words. They define definition modeling as the task of generating a definition sequence (D) for a given word (w^*). This could also be thought of as a more direct way for representing the semantics captured by embeddings. In that work, their proposed model is built on a recurrent neural network (RNN) based language model [61] which models the probability of the definition sequence as follows:

$$P(D|w^*) = \prod_{t=1}^{T} p(w_t|w_1, ..., w_{t-1}, w^*) \tag{2.3}$$

$$P(w_t = j|w_1, ..., w_t - 1, w^*) = Softmax(\mathbf{W}_d\mathbf{h}'_t + \mathbf{b}_d) \tag{2.4}$$

where T is the length of the definition sequence (D), and w_t is the t^{th} word of D. $Softmax$ is the softmax function applied on model weights ($\mathbf{W}_d$), model bias ($\mathbf{b}_d$), and the hidden state of the LSTM definition decoder ($\mathbf{h}'_t$).

They also improve their RNN-based model by incorporating a character-level convolutional neural network (CNN) [44] which leads to substantial improvements. In order to condition the model on the target word, the authors propose several methods for incorporating the given word into the model such as adding the word to the beginning of the definition sequence as a seed, providing the model with the word at each time step, and updating the hidden representation using the given word. For the purpose of training and evaluation, they present a dataset which contains words associated with their corresponding definitions from WordNet [63] — a lexical database of semantic relations between words developed in more than 200 languages — and GNU Collaborative International Dictionary of English (GCIDE).[4] The models proposed in this study utilize the pre-trained Word2Vec embeddings [60]. The use of single-sense word embeddings in this work, restricts their model to generating definitions only for the most frequent meaning of the given word and ignores polysemy. This is a particularly important limitation because a significant number of English words are associated with more than one sense [63].

To overcome the limitation of the first work [69] in dealing with polysemy, a number of studies have been published taking a context-agnostic approach. Yang et al. [93] propose a context-agnostic model which incorporates sememes into Chinese definition generation. Sememes are defined as minimum semantic units of word meanings, and usually the meaning of each word sense is composed of several sememes. For example, the meaning of the word *hotel* is composed of five sememes, which are *place*,

[4] `http://gcide.gnu.org.ua/`

tour, *eat*, *recreation*, and *reside*. In this work, the authors first construct a Chinese dataset containing triples of the target word, sememes and a definition for a specific sense of the target word, where the sememes are annotated with HowNet[5] [98], and the definitions are annotated with Chinese Concept Dictionary (CCD) [96]. Then, to incorporate sememes into the definition generation task, they propose two models within the encoder-decoder framework. The first model employs adaptive attention mechanism [53] in a LSTM-based encoder-decoder architecture. In their second model, however, they totally replace RNN connections with transformers [89]. This fully attention-based model allows for more parallelization and better performance. At test time, their models are able to generate a definition for a specific sense of the target word by encoding the sememes corresponding to that given sense. Although, by incorporating sememes, they are able to address polysemy in a context-agnostic approach, their model suffers from a noticeable limitation. In their proposed model, sememes which are the core information to sense-specific definition generation are provided by HowNet which uses manual annotation and covers a limited number of words. Obviously, it does not include sememes for words that have recently emerged — e.g. the word *felfie* which refers to a photo people take of themselves and their family — or words that have recently taken on new meanings — e.g. the word *tweet* which recently has taken another meaning related to the social media service Twitter in addition to its traditional meaning. On the other hand, our model, to address polysemy, utilizes multi-sense embeddings which are produced by semi-supervised methods without the need for any manual annotation. To include multi-sense embeddings for newly-emerging words, the multi-sense embedding method just needs to be re-trained on a recent corpus (i.e. Wikipedia[6] or Google News[7]).

[5]HowNet is a common-sense knowledge base maintaining inter-conceptual and inter-attribute relationships of concepts as implying in lexicons of the Chinese and their English equivalents.

[6]`https://www.wikipedia.org`

[7]`https://www.news.google.com`

In another context-agnostic study, to overcome the limitation of [69] in generating multiple definitions for polysemous words, Zhu et al. [99] propose a multi-sense model for generating definitions for the various senses of a target word. This model utilizes word embeddings and coarse-grained atom embeddings to represent senses [4], in which atoms are shared across words. In an atom embedding method, instead of learning representations for each word type, representations are learned for more general concepts and topics. During the decoding stage, the vector representation of a given word is a concatenation of the word embedding, atom embedding, part of speech embedding, and output of a CNN character-level affix detector. Then, the LSTM gates control the information flow coming from each of these sources. In this multi-sense setting, the dataset contains several definitions for each word corresponding to multiple meanings of the word. On the other hand, each word is associated with multiple atom embeddings which correspond to different senses of the word. To match sense vectors to reference definitions during training, the authors propose a neural approach utilizing Gumbel-Softmax [38], and also consider a heuristic-based approach that incorporates cosine similarity between senses and definitions. In the current study, our proposed approach to this matching is similar to their heuristic-based approach, although we explore two variations of this method. Besides, contrary to their approach, our model does not stick to word embeddings and only relies on fine-grained multi-sense embeddings. Furthermore, [99] only consider English for evaluation, whereas we consider fifteen datasets covering nine languages.

In a different context-agnostic study [7], instead of focusing on polysemy, the authors address domain-specific definition generation. Specifically, they propose a model inspired by the definition generation model proposed in [69] for generating definitions for the terms in the software domain. In their model, they propose to incorporate

some domain-specific knowledge such as co-occurrence entities and ontology information into it. In this work, entities refer to the tags that are associated with each question in Stack Overflow website,[8] and co-occurrence entities are the tags that are seen together on Stack Overflow questions. The authors also propose to use an augmented version of cross entropy loss function to force the model to reconstruct the given term from the generated definition. One of the additional sources of information they proposed to benefit from is the entity–entity association information; however, it does not prove to be useful as it does not lead to any improvements. Another source of information they incorporate into the model is the ontological category information which proves to be helpful and increases the robustness of the model. Besides, to train and evaluate their model, they construct a new dataset of the software-specific terms associated with their technical definitions and some additional information — e.g. co-occurrence entities and ontology information — which is gathered from Stack Overflow forums. Employing pre-trained word embeddings, their model still lacks the ability to generate sense-specific definitions for multiple meanings of polysemous words as the word embeddings utilized in their work do not capture multiple senses of each word type separately.

2.3.2 Context-aware methods

Inspired by the fact that words are defined by the contexts they are used in [60], a number of studies have proposed context-aware models considering the context the target word appears in for definition generation. The main aim of these approaches is to address homonymy and polysemy. Homonymy and polysemy are two concepts for lexical ambiguity of language which imply the presence of multiple related and unrelated meanings for a single word, respectively.

[8] `https://stackoverflow.com`

As the first context-aware study, Ni and Wang [68] propose a neural network sequence-to-sequence model for defining newly emerging non-standard English expressions considering the context they are used in. Their proposed model consists of two encoders and one decoder. In their proposed model, the encoders first encode the contextual information as well as the character-level structure of the given phrase. Then, a decoder is trained on the linear combination of the encoders to predict a definition for the given phrase. The results demonstrate the superiority of the dual-encoder model in terms of BLEU score compared to the models with one encoder. Although they have taken the contextual information into account, all senses of a word are still represented using a single dense vector, which may hinder their model from dealing with polysemy.

To specifically address polysemy and homonymy, a number of studies have proposed context-aware approaches to definition modeling. In [28], the authors propose two RNN-based models for disambiguating word senses in the definition modeling task using the context in which a word appears. Specifically, they utilize the Adaptive Skip Gram model [8] and an attention mechanism which uses the context of a word being defined for disambiguation. The proposed model in this work uses the contextual information to predict a proper definition with respect to the word meaning intended in the context. Similarly, in another context-aware study [37] propose an RNN-based conditional language model benefiting from both local and global contexts — local context is defined as the explicit contextual information included in a single sentence, while global context refers to the implicit contextual information in the word embedding trained in an unsupervised manner on large-scale corpora — to generate definitions for rare or unseen words and phrases. In this work, the global contexts are captured using single-sense pre-trained word2vec word embeddings [60], while a bi-directional LSTM is used to capture the local contextual information of

a given word. A uni-directional LSTM, equipped with the attention mechanism, is then applied on the encoder output. This work, unlike the previous ones, is capable of using both global and local contexts in isolation and alongside each other. However, this model, to address polysemy, is still directly dependent on the appearance of the target word in a context. Moreover, this study only considers single-sense word embeddings in the proposed model. In fact, without having the local contextual information, this model is not capable of generating definitions for multiple senses of the target word.

One of the interesting characteristics of word embeddings is that words have large values in specific dimensions of their sparse representations which correspond to specific semantic concepts [26, 85]. Benefiting from this property, Chang et al. [18], propose a context-aware model learning a mapping from the dense vector representation of a word to a sparse vector representation of higher dimension, in which each dimension represents specific concepts or senses. Then, in a component called Mask Generator, they take a dot product of this sparse vector and the encoded local context of the target. This step assigns higher attention weights to the dimensions corresponding to the word sense that is meant in the given context. Eventually, after the sense disambiguation step, they train a language model with a two-layer GRU [19] which is conditioned on the word embedding and the context masked vector to generate sense-specific definitions. Their proposed model demonstrates an improvement in terms of BLEU score compared to context-agnostic and context-aware baselines.

As mentioned in Section 2.1.3, contextualized embeddings like ELMo [73] and BERT [23] have outperformed traditional single context-independent word embeddings and made significant improvements on NLP tasks [2, 51, 15, 55]. Zhang et al., to utilize contextualized embeddings in definition modeling, propose a context-aware architecture

based on encoder-decoder to generate definitions and usage examples for a given word with respect to the context the word appears in [97]. Their proposed model first employs BiGRU with a max pooling layer [21] to encode the context of the given word. Then, using Scaled Dot-Product Attention [89] between the word embedding and the encoded context, it produces a context-aware semantic representation for the target word in the given context, which is then fed to a two layer GRU decoder as the input at each time step. ELMo embeddings are used to obtain the contextualized embedding of the target word which is given to the decoder as the initial hidden state. In this framework, to address usage modeling, they also propose two multi-task sequence to sequence models [54] to combine definition modeling and usage modeling sharing the representations at different levels. Usage modeling, similar to definition modeling, is a dictionary-related task in which a model is trained to generate dictionary-style usage examples for any given word. In their multi-task models, they use two separate decoders to generate definition and usage examples, both of which get the input vectors from a shared encoder. Their proposed models achieve state-of-the-art performance on both definition and usage modeling at the time of the writing.

In a different study, the authors propose to reformulate the task of definition modeling from natural language generation (NLG) to classification [17]. In their new formulation, their model learns a mapping between the semantic space of contextualized word embeddings and the space of word definition embeddings. In this study, they use the pre-trained transformer-based universal encoder [16] to encode both definitions and contexts. In addition to the employment of the sentence encoder, they also propose to employ ELMo and BERT to obtain context-dependant vector representations for the given word. Next, a 7-layer multi-layer perceptron is applied on the contextualized word embedding to translate it to a point in the space

of definition embeddings. Finally, during the inference stage, given a target word and its context, a proper definition is retrieved from the top k nearest neighbors of the predicted embedding in the definition embedding space. This work, despite its novel reformulation of definition modeling from NLG to classification and good performance over prior work, is restricted only to previously-existing definitions. That is, for a new word with a new meaning, this model is not able to predict the correct definition as it does not have a relevant definition for that meaning in its database.

In [69], definition modeling was introduced as a word-to-sequence task in which given the embedding of a word, a proper definition is generated. Obviously, this modeling is restricted to generating a single definition for the given word ignoring semantic ambiguity of the given word. Studies like [28, 37, 18] then propose context-aware models to disambiguate the given word with respect to the given context. In a different context-aware study [59], the authors argue that sequence-to-sequence is a more natural way of formulating definition modeling instead of word-to-sequence. They believe during the definition generation, the decoder needs to have access to the target word and all the tokens accompanying the target word in a context, rather than only one contextualized word embedding for the target word. In this work, they propose two simple approaches to mark the tokens in the given sequence as the target word or context word. To evaluate their proposed formulation, they employ a transformers-based definition generator [89]. Their model demonstrates significant improvements compared to prior work.

All the works described in this section, despite their good performance, require local contextual information of the target word to generate multiple sense-specific definitions for multiple senses of polysemous words. In this setting, in order to generate definitions corresponding to all possible senses of a polysemous word, you need to

feed the model with usage examples for each sense of the target word. The main idea of our work is to propose a context-agnostic definition generation model which tries to generate sense-specific definitions for all existing senses of a given word attested in a corpus, without the need for providing usage examples. Our proposed model incorporates multi-sense embeddings into a definition generation model to achieve this aim.

Chapter 3

Proposed Model

Definition modeling is a recently-introduced NLP task which aims to predict the likelihood for any sequence of tokens as a candidate definition for a given target. In this section, we describe our two proposed multi-sense definition generation models which take a word and its sense-embeddings as input and generate multiple definitions, one for each of different recognized senses of the given word, as output.

The proposed multi-sense definition generation models in this study, unlike most of the previous works [28, 37, 59], are context-agnostic which means that the definitions of a word are predicted regardless of the context the word appears in. The intuition behind proposing context-agnostic models in this study is that the number of senses a word can convey in various contexts is not absolute. That is, the borders between different meanings each word can imply are to some extent vague. For instance, in two example sentences *I will go to the bank to deposit some money to my account* and *the bank of Canada has changed some of its privacy policies*, one may consider the word *bank* to imply the same financial meaning, while one may see different meanings for this word in a more meticulous perspective: *a financial building* and *a financial institution*. As a result, a context-aware model may end up producing

possibly different definitions for each usage of the given word in different sentences. Subsequently, for a word type seen in n sentences, we may end up having 1 to n different definitions, which is not what we see in dictionaries — i.e. for the usages of the target word in multiple similar contexts, only one definition is listed. Therefore, instead of considering the local contexts in our model, we propose to rely on the global context captured by the sense embedding methods from large corpora to generate sense-specific definitions.

In this study, we propose to extend the single-sense model proposed by Noraset et al. in [69] to a multi-sense definition generation model. Therefore, we build our models on the basis of this single-sense model which we refer to as the base model in the remainder of this thesis.

3.1 Single-Sense Base Model

The base model is basically a language model initialized with word embeddings which is conditioned on a word being defined (w^*) to learn the definition ($D = [w_1, ..., w_t, ..., w_T]$) of the given word. The probability of the tth word of the definition sequence D is calculated based on the previous words, $[w_1, ..., w_{t-1}]$, in the definition as well as the word being defined w^* (Equation 3.1) where T is the length of the definition sequence (D), and w_t is the tth word of D. The probability distribution is estimated by a softmax function (Equation 3.2).

$$P(D|w^*) = \prod_{t=1}^{T} p(w_t|w_1, ..., w_{t-1}, w^*) \tag{3.1}$$

$$P(w_t = j|w_1, ..., w_t - 1, w^*) = Softmax(\mathbf{W}_d \mathbf{h}'_t + \mathbf{b}_d) \tag{3.2}$$

$Softmax$ is the softmax function applied on model weights ($\mathbf{W}_d$), model bias ($\mathbf{b}_d$),

and the hidden state of the LSTM definition decoder ($\mathbf{h}'_t$).

The base model is composed of a recurrent neural network (RNN) — long short term memory (LSTM) is chosen as the RNN unit — for learning the sequence of the words appearing in the definitions and a fully-connected layer (FC layer) mapping hidden units to the vocabulary words whose output is given to a softmax for the calculation of the probabilities. The target word to be defined is given to the network as the initial hidden state, and a special token (<BOD>) is given as the seed input token to the network as depicted in Fig. 3.1. At each time step, the index of the predicted definition word (v^n) is mapped into the word's embedding to be fed into the model at the next time step.

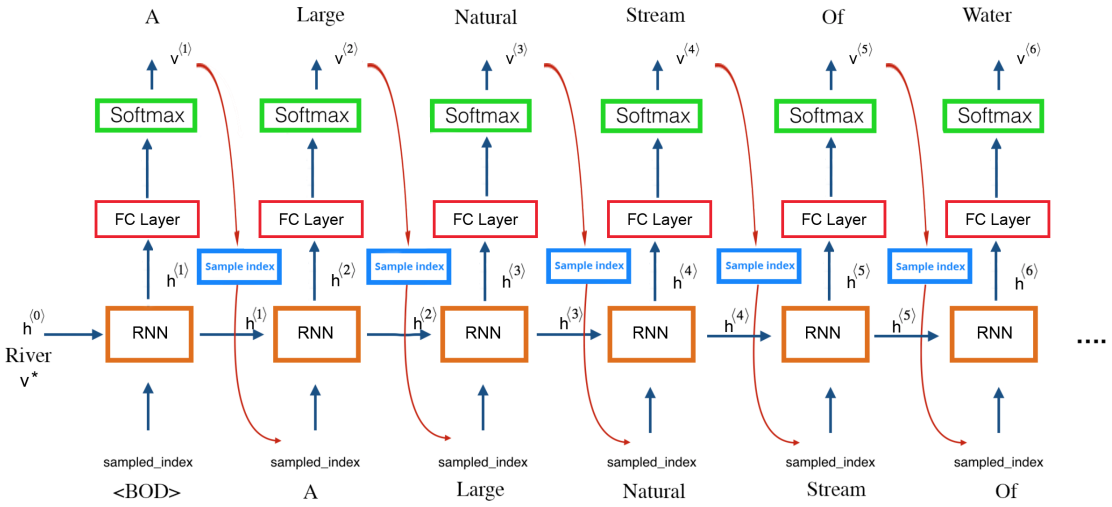

Figure 3.1: An illustration of the recurrent architecture of the single-sense definition generation model.

The hidden state of the RNN unit at each time step (h_t) in the base model is calculated as follows:

$$h_t = g(v_{t-1}, h_{t-1}, v^*) \tag{3.3}$$

where g is a non-linear function for the network, v_t denotes the vector representation of the tth token of the definition sequence, and v^* indicates the vector representation of the target word being defined. Moreover, in order to let the model pay different attentions to the target word when generating definition tokens — including words carrying semantic information and non-semantic words like function words and stop words — the output of the recurrent unit is updated with a GRU-like update function as follows:

$$z_t = \sigma(W_z[v^*; h_t] + b_z) \tag{3.4}$$

$$r_t = \sigma(W_r[v^*; h_t] + b_r) \tag{3.5}$$

$$\tilde{h_t} = tanh(W_h[(r_t \odot v^*); h_t] + b_h) \tag{3.6}$$

$$h_t = (1 - z_t) \odot h_t + z_t \odot \tilde{h_t} \tag{3.7}$$

where $[a; b]$ denotes concatenation between two vectors a and b, σ is the sigmoid function used as a non-linearity, and $\odot$ represents element-wise vector multiplication. z_t is the output gate controlling how much of the output of the RNN unit should change, and r_t is a reset gate controlling how much information from the target word v^* should be passed in. h_t presented in Equation 3.3 is finally updated as shown in Equation 3.7.

In addition to the RNN unit described above, a character-level convolutional neural network (CNN) is employed to capture sub-word features. The reason behind utilizing this CNN unit is to enable the model to better deal with out-of-vocabulary

(OOV) words — i.e. words for which no vector representation is learned in the given pre-trained word embeddings. It is hypothesized that sub-word features like words' roots, prefixes, and suffixes can provide a good estimation of the meaning of the given OOV word.

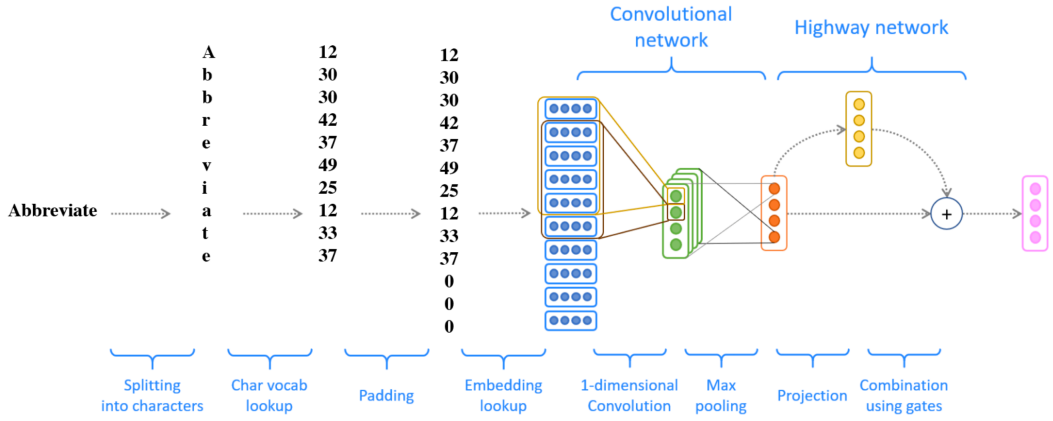

Figure 3.2: An illustration of the character-level CNN unit employed in the single-sense definition generation model [95].

The employed character-level CNN in this model is inspired by the character-level language model proposed in [44]. As could be seen in Fig. 3.2, each word is first split into its constituent characters, then each character is mapped into its index in a character dictionary. Next, in order to have fixed-size vectors for every given word, zero padding technique is applied. The character indices then are mapped into randomly-initialized low-dimensional dense vector representations such that each character is represented by a low-dimensional dense vector. In the CNN unit, different kernels of varying sizes are applied on the word characters to capture n-gram features from the given word. A max pooling layer is used to decrease the dimensionality of the obtained feature vectors. Finally, a highway network is employed to get the final sub-word vector representation for the given word. Highway network is an approach utilized to optimize networks and increase their depth. Highway networks, inspired

by LSTM recurrent neural networks, use trainable gating mechanisms to regulate information flow.

3.2 Multi-Sense Models

The base model described in the previous section utilizes a pre-trained word embedding method which assigns a single vector representation to each word type regardless of the different multiple meanings the word can imply in various contexts. This, in turn, limits the ability of the definition model in generating definitions for different senses of a given polysemous word.

In order to extend the base model to be able to generate multiple sense-specific definitions for the polysemous words, we propose to utilize multi-sense embeddings. A multi-sense embedding method, instead of learning one vector representation for each word type, learns separate vector representations for each recognized sense of a word type. In order to differentiate between multiple different senses a word implies, it uses the contexts the word has appeared in throughout the training corpus. Among existing methods to learn multi-sense embeddings, some methods automatically detect the number of different senses a word may imply in different contexts, which are referred to as non-parametric methods [8, 66]. On the other hand, some multi-sense embedding methods like MUSE [48] are parametric which means that they assume a fixed number of different senses for all words.

As described in Section 2.1.3, contextualized word embeddings are very recent successful methods to learn multiple vector representations for each word type, which seems to be a good candidate to enable definition generation models to generate multiple sense-specific definitions for polysemous words. However, we do not believe

they can be a good solution to the problem as they produce embeddings based on the local contexts which could be utilized in context-aware definition generation models. Since multi-sense embedding methods, as opposed to contextualized embedding methods, use global contextual information obtained from a huge training corpus, we propose to use them in our context-agnostic multi-sense definition generation model.

The datasets that we use in our experiments to train and evaluate the models are monolingual dictionaries containing varying numbers of definitions associated to each word (one for monosemous words and more than one for polysemous words). On the other hand, in the employed pre-trained multi-sense embeddings, multiple vector representations are assigned to each word — one to each recognized sense of the word. Note that the number of existing reference definitions available in the dataset for a given word is not necessarily equal to the number of vector representations learned by the multi-sense embedding method for the same word.

Therefore, in order to utilize multi-sense embeddings in a definition generation model, we need a function to map the trained sense-specific vector representations of a word to the corresponding reference definitions provided for the word in the datasets. To this aim, we propose to employ cosine similarity function to calculate pair-wise similarity between multiple sense vectors and reference definitions provided for each word in the dataset. Then, the sense–definition pairs with the highest similarity are chosen as the training instances for the model. The cosine similarity function calculating the similarity between two given vectors p and q is as follows:

$$Similarity(p,q) = cos\theta = \frac{p \cdot q}{\|p\|\|q\|} = \frac{\sum_{i=1}^{n} p_i q_i}{\sqrt{\sum_{i=1}^{n} {p_i}^2}\sqrt{\sum_{i=1}^{n} {q_i}^2}} \tag{3.8}$$

To associate sense vectors of a target word with corresponding reference definitions, we consider two approaches employing cosine similarity function: definition-to-sense (Def2Sense) and sense-to-definition (Sense2Def). For both approaches we require a representation of definitions. As described in Section 2.2, there are various methods to learn vector representations for documents (sequences of words). In this study, we propose to represent a definition as the average of its word embeddings — similar to CBOW method [60] — after removing stopwords. For each word in the training data, we then calculate the pairwise cosine similarity between its sense vectors and definitions. For Def2Sense, each definition is associated with the most similar sense vector for the corresponding word. For sense-to-definition, on the other hand, each sense is associated with the most similar definition. For both approaches, the selected Sense2Def pairs form the training data.

Note that these approaches to pairing senses and definitions are only used to create training instances. At test time, to generate definitions for a given target word, each sense vector for the target word is fed to the definition generation model, which then generates one definition for each of the target word's sense vectors. Note that as we intend to evaluate the ability of the proposed multi-sense models in generating multiple definitions for unseen words, the words that the models are evaluated on at test time are not among the words that the models were trained on.

The overall architecture of the proposed multi-sense definition generation model is illustrated in Fig. 3.3.

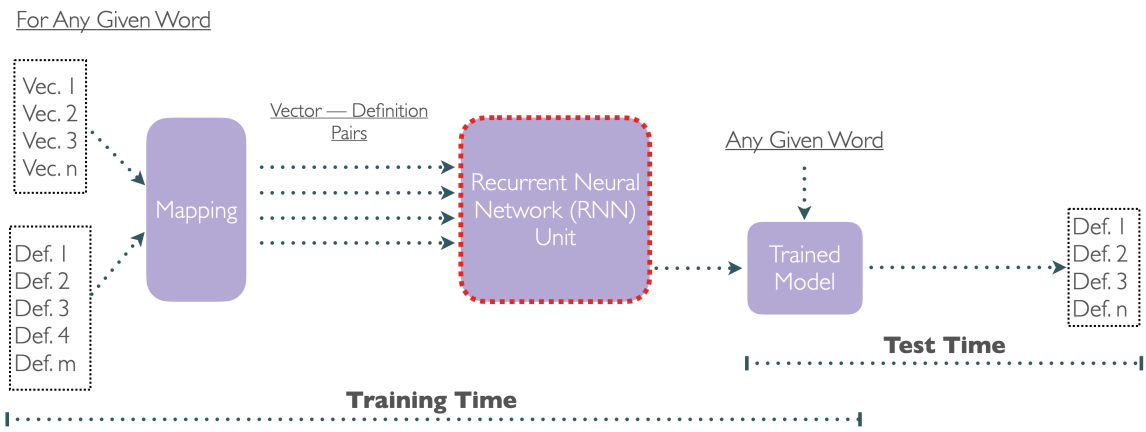

Figure 3.3: An illustration of the overall architecture of the proposed multi-sense model.

Chapter 4

Experimental Setup

In this chapter, we describe the datasets, word and sense embeddings, evaluation metrics, model setup, and the baseline used in our experiments.

4.1 Datasets

In this work, we conduct a multi-lingual study of definition modelling. To this aim, we extract monolingual dictionaries for nine languages covering several language families, from three different sources: Wiktionary,[1] OmegaWiki,[2] and WordNet [63].

Wiktionary is a free collaboratively-constructed online dictionary for many languages. A disappointing point about Wiktionary is that the structure of Wiktionary pages is not consistent across languages. Therefore, extracting word–definition pairs from Wiktionary pages for a given language requires a carefully-designed language-specific parser, which moreover requires some knowledge of that language to build. We therefore use publicly-available Wiktionary parsers. We use WikiParsec for En-

[1] `https://en.wiktionary.org`
[2] `http://www.omegawiki.org`

glish, French, and German,[3] and Wikokit for Russian,[4] to extract word–definition pairs for these languages. English, French, German, and Russian are the only languages for which we have been able to find publicly-available parsers.

Another resource that we use to construct the desired dictionaries for our experiments is OmegaWiki. OmegaWiki, like Wiktionary, is a free collaborative multilingual dictionary. A significant advantage of OmegaWiki over Wiktionary is that in OmegaWiki, the data is stored in a relational database. This makes the process of extracting words and definitions much easier to automate without the need for language-specific parsers. However, the size of the dictionaries is smaller compared to those of Wiktionary. Therefore, we extract the word–definition pairs from OmegaWiki for English, Dutch, French, German, Italian, and Spanish — the six languages with the largest vocabulary size in OmegaWiki — using the BabelNet Java API [65].

The third resource which we use to build the monolingual dictionaries for our experiments is WordNets [63]. We only use WordNets for which the words and definitions are in the same language. Many WordNets published for languages other than English only map words of the target language to the definitions and synsets of the English WordNet. That is, they do not offer definitions for the words in the target language. Therefore, we cannot use them to build suitable datasets for our experiments in definition modeling. To extract the word–definition entries from English [63], Italian [6], and Spanish [27] WordNets, we again use the BabelNet Java API [65]. We separately extract word–definition pairs from Greek [84] and Japanese [36] WordNets.

[3] `https://github.com/LuminosoInsight/wikiparsec`
[4] `https://github.com/componavt/wikokit`

Properties of the extracted datasets are shown in Table 4.1. The number of words defined in all dictionaries varies between 11500 to 20000. As could be seen, a significant difference between the dictionaries extracted from the existing sources is the variance of the number of definitions per each word. In OmegaWiki, the variance ranges between 0.36 to 0.86, which means the number of definitions associated to each word does not vary too much. However, this value goes to the range of 1.05 to 7.73 in Wiktionary dictionaries. In WordNet dictionaries, the variance is still higher than that of OmegaWiki, but to a lower extent. It is worth mentioning that the machine that we have access to for our experiments is equipped with an NVIDIA V100 Volta GPU with 16 GB of memory, while the datasets we construct for the experiments are quite large and require more than 16 GB of GPU memory. As such, due to this computational limitation, we have to down-sample the extracted datasets. To do so, we randomly choose a subset of the target words to be defined from the datasets. The down-sampling ratio, depending on the size of the original datasets, varies between 0.5 to 0.75. Note that the properties reported in Table 4.1 correspond to the down-sampled datasets.

The datasets are split into training, development, and test sets with portions of 80%, 10%, and 10%, respectively. We make sure that during the split, all the definitions for any given word are included in only one of the three sets as we intend to explore the ability of the models in generating the definitions of the words that were not previously seen during the training phase.

4.2 Word and Sense Embeddings

Following Noraset et al. [69], we use word2vec embeddings [60] in the singe-sense definition generation model (i.e., the base model described in Section 3.1).

Language	# Words	Proportion polysemous	Average	Variance
		Wiktionary		
English	17000	0.27	1.73	7.73
French	20000	0.26	1.39	1.14
German	16000	0.26	1.48	1.50
Russian	15000	0.17	1.33	1.05
		OmegaWiki		
Dutch	13093	0.18	1.24	0.36
English	17000	0.20	1.33	0.86
French	15869	0.17	1.26	0.59
German	13338	0.12	1.17	0.34
Italian	18351	0.21	1.33	0.68
Spanish	17000	0.19	1.30	0.64
		WordNet		
English	20000	0.18	1.44	2.81
Greek	11517	0.26	1.50	1.47
Italian	16290	0.22	1.37	0.79
Japanese	20000	0.30	1.47	0.893
Spanish	18934	0.12	1.21	0.71

Table 4.1: Properties of the extracted Wiktionary, OmegaWiki, and WordNet dictionaries for nine languages. Proportion polysemous presents the portion of polysemous words in each dictionary. In the average column, the average of the number of definitions per word is given for each dictionary. In the variance column, the variance of the number of definitions per word is shown for each dictionary.

For the proposed multi-sense definition generation models, we utilize AdaGram multi-sense embeddings [8]. AdaGram is a non-parametric Bayesian extension of skip-gram [62] which learns a variable number of sense vectors for each word, unlike many multi-sense embedding models which learn a fixed number of senses for every word. Note that although in our experiments, we choose AdaGram method to train multi-sense embeddings, any multi-sense embedding method could potentially be used.

It is worth mentioning that we initially also considered MUSE multi-sense embeddings [48], but as it performed poorly compared to AdaGram [8], we did not continue the experiments with MUSE. Table 4.2 shows some properties about the senses recognized by AdaGram for nine languages and MUSE for English. As could be seen, the similarities between the senses detected by MUSE for each word are much higher than the similarities between the senses recognized by AdaGram. This observation can justify the reason of the poor performance of MUSE compared to that of AdaGram in our experiments. What we conclude from these results is that AdaGram has been more successful in differentiating between multiple different senses of polysemous words and learning corresponding embeddings. For this reason, we only report the results of our experiments with AdaGram [8] in Chapter 5.

For each language, word2vec and AdaGram embeddings are trained on the most recent Wikipedia dumps as of January 2020.[5] We extract plain text from these dumps, and then pre-process and tokenize the corpora using tools from AdaGram,[6] modified for multilingual support, except in the case of Japanese where we use the Mecab tok-

[5] `https://dumps.wikimedia.org`

[6] `https://github.com/sbos/AdaGram.jl/blob/master/utils/tokenize.sh`

Language	Avg. Sim.	Min. Sim.	Avg. Num.
	AdaGram		
Dutch	0.25	0.21	1.91
English	0.27	0.22	2.58
French	0.28	0.24	1.90
German	0.26	0.22	2.41
Greek	0.24	0.19	2.90
Italian	0.26	0.21	2.25
Japanese	0.25	0.17	4.19
Russian	0.23	0.17	2.98
Spanish	0.25	0.21	2.17
	MUSE		
English	0.81	0.80	2.99

Table 4.2: Some statistical information about the sense embeddings trained by AdaGram and MUSE methods. Columns Avg. Sim., Min. Sim., and Avg. Num. represent average of similarities between senses of each word, average of minimum similarities between senses of each word, and average of number of senses per word, respectively.

enizer.[7] The resulting corpora range in size from roughly 86 million tokens for Greek to 3.7 billion tokens for English (presented in Table 4.3). The same pre-processing and tokenization is also applied to the datasets of words and definitions extracted from dictionaries.

Language	DE	EL	EN	ES	FR	IT	JA	NL	RU
# Tokens	1.2B	86M	3.7B	811M	1.1B	675M	586M	355M	773M

Table 4.3: The number of tokens of the training corpora

We train word2vec embeddings [60] using Gensim with its default parameters.[8] We also use the default parameter settings for AdaGram [8] as these default parameters are demonstrated in the original paper to lead to the best performance of the model.

[7] `https://github.com/jordwest/mecab-docs-en`

[8] `https://radimrehurek.com/gensim/`

To obtain representations for words, as opposed to senses, from AdaGram sense embeddings, as required to form representations for definitions (Section 3.2), we take the most frequent sense vector of each word (as indicated by Adagram) as the representation of the word itself.

4.3 Evaluation Metrics

BLEU [71] has been widely used for evaluation in prior work on definition modelling [69, 37, 68]. While BLEU is appropriate for evaluation of single-sense definition generation models, it does not capture the ability of a model to produce multiple definitions corresponding to different senses of a polysemous word. We therefore also consider a recall-based variation of BLEU, known as rBLEU, in which the generated and reference definitions are swapped [99], i.e., the overlap of a reference definition is measured with respect to the generated definition(s). For each target word, we calculate rBLEU as the average rBLEU score for each of its reference definitions (for both single and multi-sense models). In addition to precision-based BLEU, and recall-based rBLEU, we report the harmonic mean of BLEU and rBLEU, referred to as fBLEU (equation 4.1).

$$fBLEU = \frac{2 * BLEU * rBLEU}{BLEU + rBLEU} \tag{4.1}$$

4.4 Model Setup

For choosing the overall architecture and hyper-parameters of the models, we follow [69]. All of the definition generation models in this study utilize a two-layer LSTM as the RNN unit with 300 units in each level and a character-level CNN with kernels of length 2-6 and size 10, 30, 40, 40, 40 with a stride of 1 for detecting the

affixes. The optimization algorithm used to train the language models is Adam [8] with the learning rate of 0.001. During the definition generation phase in test time, we sample tokens at each time step from the predicted probability distribution of words with a temperature of 0.1. In order to remove the effect of chance from our results, we do this process 10 times and generate 10 definitions at each definition generation step. Finally, we report the averaged BLEU, rBLEU, and fBLEU scores.

4.5 Baseline

The baseline against which we compare our model is the single-sense model proposed by Noraset et al. [69] which utilizes word2vec embeddings [60]. The reason we choose this work as the baseline of our study is twofold. First, this study is the first work introducing the definition modeling task and is considered as a standard baseline by most of prior work on definition modeling. Second, most of the recent works on definition modeling have conducted context-aware studies [68, 28, 37, 59, 17], and the single-sense model proposed by Noraset et al. [69] is one of the few context-agnostic approaches.

Chapter 5

Results

In this section, we present experimental results comparing the performance of the proposed multi-sense definition generation models described in Section 3.2 (Def2Sense and Sense2Def) against the single-sense base model [69] described in Section 3.1. The performance of the models is first evaluated quantitatively using variations of the BLEU score (Section 5.1), and is then evaluated qualitatively (Section 5.2).

5.1 Quantitative Results

The performance of the proposed multi-sense models on OmegaWiki, Wiktionary, and WordNet datasets is quantitatively compared against that of the base (single-sense) model in Table 5.1 using variations of BLEU score. What is most remarkable in the presented results is that across all fifteen datasets covering nine languages, a multi-sense model — i.e., Sense2Def or Def2Sense — has outperformed the single-sense baseline in terms of rBLEU and fBLEU while BLEU is not substantially impacted. The results presented in Table 5.1 show us that not only rBLEU and fBLEU but also in many cases the original BLEU score has improved by incorporating multi-sense embeddings on Wiktionary datasets. This means not only are the proposed

Table 5.1: BLEU, rBLEU, and fBLEU for the single-sense definition generation model (base) and the proposed multi-sense models using Sense2Def (S2D) and Def2Sense (D2S) for each dataset. The best result for each evaluation metric and dataset is shown in boldface.

Lang.	Model	OmegaWiki			Wiktionary			WordNet		
		BLEU	rBLEU	fBLEU	BLEU	rBLEU	fBLEU	BLEU	rBLEU	fBLEU
DE	base	12.12	11.55	11.83	11.35	08.80	09.91	–	–	–
	S2D	12.43	16.26	14.09	**15.00**	15.82	**15.40**	–	–	–
	D2S	**12.44**	**16.83**	**14.31**	14.07	**16.54**	15.21	–	–	–
EL	base	–	–	–	–	–	–	**13.21**	12.06	12.61
	S2D	–	–	–	–	–	–	12.44	12.85	12.64
	D2S	–	–	–	–	–	–	13.08	**13.63**	**13.35**
EN	base	14.74	14.32	14.53	20.21	16.88	18.40	13.78	12.77	13.26
	S2D	14.23	16.02	15.07	18.88	16.99	17.89	12.85	13.09	12.97
	D2S	**15.22**	**17.80**	**16.41**	**21.49**	**19.78**	**20.60**	**13.84**	**14.84**	**14.32**
ES	base	**17.68**	17.70	17.69	–	–	–	**26.46**	24.69	25.54
	S2D	16.52	19.00	17.67	–	–	–	25.80	**28.14**	**26.92**
	D2S	17.54	**20.28**	**18.81**	–	–	–	25.68	27.97	26.78
FR	base	**12.58**	12.66	12.62	63.48	59.87	61.62	–	–	–
	S2D	11.70	14.26	12.85	63.56	60.00	61.73	–	–	–
	D2S	11.94	**14.82**	**13.23**	**64.12**	**60.41**	**62.21**	–	–	–
IT	base	**12.29**	11.93	12.11	–	–	–	21.33	20.65	20.98
	S2D	11.43	13.61	12.43	–	–	–	20.35	23.67	21.88
	D2S	11.74	**13.95**	**12.75**	–	–	–	**21.96**	**25.10**	**23.43**
JA	base	–	–	–	–	–	–	10.13	08.50	09.24
	S2D	–	–	–	–	–	–	**11.53**	**11.96**	**11.74**
	D2S	–	–	–	–	–	–	09.42	09.37	09.39
NL	base	14.37	14.04	14.20	–	–	–	–	–	–
	S2D	13.49	15.88	14.59	–	–	–	–	–	–
	D2S	**14.46**	**17.07**	**15.66**	–	–	–	–	–	–
RU	base	–	–	–	47.04	46.04	46.53	–	–	–
	S2D	–	–	–	46.24	46.69	46.46	–	–	–
	D2S	–	–	–	**47.52**	**48.09**	**47.80**	–	–	–

models able to generate multiple definitions for different senses of a polysemous word (higher rBLEU score) but also generate definitions of higher quality (higher BLEU score).

rBLEU is a recall-based evaluation metric that indicates the extent to which the reference definitions are covered by the generated definitions. A multi-sense definition generation model — which produces multiple definitions for a target word — is therefore particularly advantaged compared to a single-sense model — such as the base model — which produces only one, with respect to this metric. Indeed, we see that for every dataset, both Sense2Def and Def2Sense, outperform the base model in terms of rBLEU. BLEU, on the other hand, is a precision-based metric that indicates whether a generated definition contains material present in the reference definitions. The improvements of the multi-sense models over the base model with respect to rBLEU do not substantially impact BLEU — as observed by the overall higher fBLEU obtained by the multi-sense models.

Focusing on fBLEU, we observe that Def2Sense often performs better than Sense2Def. The number of sense vectors learned by Adagram for a given word is on average higher than the number of reference definitions available for that word, for every dataset. We hypothesize that the poor performance of Sense2Def relative to Def2Sense could therefore be due to sense vectors being associated with inappropriate definitions.

Through a thorough inspection on the results presented in Table 5.1, we may notice the high relative performance of the models trained on French Wiktionary compared to other dictionaries. Through a manual analysis of the word–definition pairs in this dictionary, we have noticed that French Wiktionary is a very easy dataset to learn compared to other dictionaries. Most of the words defined in this dictionary

are various inflected forms of another base word. Inflection is a process in linguistic morphology in which changes are made to the base form of a word to form new words expressing different grammatical meanings (e.g. the word *colorful* is an adjective which is formed by adding the suffix *ful* to the base word *color* which has a noun meaning). As such, most of the definitions have some common phrases (i.e. *participe passé masculin singulier du verbe* which means *past participle masculine singular of verb*). This similarity in the definition sequences makes the dataset much easier to learn compared to other datasets.

5.2 Qualitative Results

In addition to the quantitative evaluation, we also qualitatively compare the proposed model against the single-sense model. In Table 5.2, the definitions generated by the single-sense model (baseline) and the def2sense multi-sense model for some example words are presented.

As could be seen, for the word *state*, the base model generates three definitions: (1) *a state of a government*, (2) *to make a certain or permanent power*, and (3) *to make a certain or administrative power*. In contrast, the multi-sense model generates the following three definitions, which appear to capture a wider range of the usages of the word *state*: (1) *a place of government*, (2) *a particular region of a country*, and (3) *a particular place of time*. Looking at the definitions generated by both models, we may notice that the proposed model has been able to generate definitions corresponding to different senses the word *state* may imply in different contexts. On the other hand, only one of the definitions generated by the base model seems to be correct (*a state of a government*), while the two other ones (*to make a certain or*

permanent power and *to make a certain or administrative power*) do not seem to be correct definitions for the target word *state*.

Baseline	Multi-sense Model
State	
a state of a government	a place of government
to make a certain or permanent power	a particular region of a country
to make a certain or administrative power	a particular place of time
Memory	
a recollection	a memorial
a recollection	a record of a computer program
the act of remembering	the act of recalling
Obstacle	
a difficulty	to be a hindrance to
a difficulty or difficult effect	a hindrance
a difficulty or difficult act	a small piece of wood or metal

Table 5.2: The qualitative comparison of the models' generated definitions for some example words.

The definitions generated by the base model and the multi-sense model for the word *memory* are both presented in Table 5.2. As could be seen, the multi-sense model has successfully generated definitions for two different meanings the word *memory* can imply in various contexts: *a memorial* and *a record of a computer program*. However, both models have mispredicted a verbal sense for the word *memory* by generating the definitions *the act of remembering* and *the act of recalling*. We argue that this misprediction is related to part-of-speech (POS) which neither of the models have explicit information about when generating a definition for a given word. We hypothesize that by incorporating the POS tags in definition modeling, we may improve this sort of mispredictions. Further, the reason of the appearance of a repetitive definition (*a recollection*) among the definitions generated by the base model

gets back to the approach we use for having the base model generate multiple definitions. To do so, we simply give the target word to the base model as input multiple times to get multiple definitions.

Finally, the improvement of the multi-sense model over the base model is evident in the definitions generated by both models for the word *obstacle*. The two last definitions generated by the multi-sense model (*a hindrance* and *a small piece of wood or metal*) cover a conceptual and a physical meaning of the word *obstacle*, respectively. On the other hand, the definitions generated by the base model only covers one of the senses of the target word (*a difficulty*).

Overall, considering both quantitative results (Table 5.1) and qualitative results (Table 5.2), the best results are obtained using a multi-sense model — i.e., sense-to-definition (Sense2Def), or definition-to-sense (Def2Sense), for every dataset. These results indicate that a multi-sense model is able to generate definitions that better reflect the various senses of polysemous words than a single-sense model, without substantially impacting the quality of the individual generated definitions.

Chapter 6

Conclusion

Natural language processing (NLP) is a sub-field of computer science and artificial intelligence which has attracted much attention in the recent years. The goal of the algorithms developed in NLP is to enable computers to understand natural (human) language. NLP algorithms typically deal with textual data which can be the transcript of speech or some written text. In NLP, different tasks have been defined such as sentiment analysis, machine translation, document classification, etc. Definition modelling is a recently-introduced NLP task which aims to generate dictionary-style definitions for a given word. For instance, given the word *river*, a trained definition generation model is expected to generate a proper sentence defining *river* such as *a natural stream of water*.

The main research question of this study is *do multi-sense embeddings enable definition generation models to generate multiple sense-specific definitions for polysemous words?* Most of the prior work on definition modeling does not account for polysemy — i.e., words can have multiple meanings depending on the context in which they are used — or has done so by considering the context in which the target word is used. For example, such a definition generation model (called context-aware) gets the word

bank and a sentence in which the word is used like *my friend and I spent an hour at the bank of St. john river*. The model then is expected to generate a definition for the given word *bank* considering its usage in the given sentence. The main goal of this study, as expressed in the research question, is to explore if employing multi-sense embeddings instead of traditional word embeddings like word2vec enables the definition generation models to produce multiple sense-specific definitions for polysemous words. In this study, as opposed to the prior work proposing context-aware approaches to definition modeling, we proposed a context-agnostic multi-sense model to employ the multi-sense embeddings in a definition generation model. A model is called context-agnostic if it produces definitions for a given target word regardless of the context the word is used in. I.e., instead of focusing on a specific given context (local context), it considers all the contexts in which the given word has appeared throughout the training corpus (global context).

To train this model, we needed to feed it with word–definition pairs. Since we utilized multi-sense embeddings instead of word embeddings, for each word we had multiple trained vectors representing senses. We also had multiple definitions associated to each word in the dataset. Therefore, we needed a mapping function to pair each of the sense vectors with the corresponding word definitions. Our proposed multi-sense model employed two similarity-based mapping approaches to associate word definitions to their corresponding sense vectors, referred to as Sense2Def and Def2Sense. After preparing the dataset for the multi-sense model training, we used an RNN based language model enhanced by a character-level CNN for subword feature extraction for definition generation.

The second research question of this study is *is the proposed multi-sense model applicable to other languages than English?* Most of the prior work targeting definition

modeling has only focused on English. In this study, we have tried to explore the performance of definition modeling on other languages to see if the proposed models has the ability of working with multiple languages. Therefore, we conducted a multi-lingual study including nine languages from several language families. As the first multi-lingual work on definition modeling, we had to prepare and construct the required resources for multiple languages. We constructed fifteen datasets for nine languages from three different online sources: Wiktionary, OmegaWiki, and WordNet. Furthermore, we trained AdaGram multi-sense embeddings [8] on Wikipedia dumps for all nine languages.

The answer to both research questions of this study based on the results of our experiments is *YES*. Specifically, our quantitative and qualitative evaluations of the models demonstrate that the proposed multi-sense models have been able to generate multiple sense-specific definitions for polysemous words. To quantitatively measure the performance of the models, we used three variations of BLEU score [71]: original BLEU, rBLEU, and fBLEU. BLEU is an evaluation measure widely-used for machine translation. It is also widely used for evaluating definition generation models. While BLEU is used to evaluate the quality of the generated word definitions, rBLEU as a recall-based measure, is utilized to assess the ability of the models in generating multiple definitions for polysemous words. fBLEU is the harmonic mean of the BLEU and rBLEU which evaluates both mentioned abilities of the models. The results of our experiments demonstrate that the proposed models have outperformed the single-sense baseline in terms of fBLEU and rBLEU on all fifteen datasets for nine languages. Code and datasets for the experiments conducted in this study are publicly available.[1]

[1] `https://github.com/ArmanKabiri/Multi-sense-Multi-lingual-Definition-Modeling`

For future work, there are different lines of research that can be taken. First, in this study we employed AdaGram multi-sense embeddings in our proposed multi-sense model. We intend to explore the possibility of employing other multi-sense embeddings. We hypothesize the performance of a multi-sense definition generation model has a direct relationship with the performance of the utilized multi-sense embeddings. In future work, we can conduct the same experiments with different multi-sense embeddings to analyze the corresponding effect on the performance of the models. Furthermore, we propose to consider the possibility of incorporating word sense induction (WSI) in multi-sense definition modeling. WSI is an open problem in NLP which concerns the automatic identification of the possible senses of each word. We believe WSI methods can be a good alternative to multi-sense embeddings in definition modeling.

Another future direction that we intend to consider is to conduct further studies on the mapping function used for associating sense vectors to word definitions. An issue with the current mapping approach that we have used in our model is that it necessarily tries to associate each word definition to one of the existing sense vectors, or each sense vector to one of the existing word definitions in def2sense and sense2def variants, respectively. In other words, it neglects the fact that for some definitions or some vector senses, there might not exist any related sense vector or word definition, respectively. The reason for this phenomenon is that the sense vectors are trained on a training corpus, while the word definitions come from a different resource which is a dictionary.

The last direction of the future work that we propose is to study definition modeling in a cross-lingual setting. All the prior work studying definition modeling has been in a mono-lingual setting. Even this thesis studies definition modeling for multiple

languages in a mono-lingual setup. That is, the given target word and the expected word definition are in the same language. Inspired by bilingual dictionaries, we intend to work on proposing a cross-lingual definition generation model which gets a target word in a language and generates a proper definition for the given word in another language. We believe cross-lingual word embeddings which map vocabularies of different languages to a same vector space can be a valuable resource for this task.

References

[1] *Twitter*, In Oxford Online Dictionary, 2020, Retrieved from `https://www.oxfordlearnersdictionaries.com/definition/english/twitter_1?q=twitter`.

[2] Alan Akbik, Tanja Bergmann, and Roland Vollgraf, *Pooled contextualized embeddings for named entity recognition*, 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) (Minneapolis, Minnesota), Association for Computational Linguistics, June 2019, pp. 724–728.

[3] Konstantinos Alexis, Vassilis Kaffes, and Giorgos Giannopoulos, *Boosting toponym interlinking by paying attention to both machine and deep learning*, Proceedings of the Sixth International ACM SIGMOD Workshop on Managing and Mining Enriched Geo-Spatial Data (New York, NY, USA), GeoRich '20, Association for Computing Machinery, 2020.

[4] Sanjeev Arora, Yuanzhi Li, Yingyu Liang, and Tengyu Ma, *Linear algebraic structure of word senses, with applications to polysemy*, Transactions of the Association for Computational Linguistics **6** (2018), 483–495.

[5] Sanjeev Arora, Yuanzhi Li, Yingyu Liang, Tengyu Ma, and Andrej Risteski, *A latent variable model approach to pmi-based word embeddings*, Transactions

of the Association for Computational Linguistics **4** (2016), 385–399.

[6] Alessandro Artale, Bernardo Magnini, and Carlo Strapparava, *Wordnet for italian and its use for lexical discrimination*, AI*IA 97: Advances in Artificial Intelligence (Berlin, Heidelberg) (Maurizio Lenzerini, ed.), Springer Berlin Heidelberg, 1997, pp. 346–356.

[7] Vidhisha Balachandran, Dheeraj Rajagopal, Rose Catherine Kanjirathinkal, and William Cohen, *Learning to define terms in the software domain*, Proceedings of the 2018 EMNLP Workshop W-NUT: The 4th Workshop on Noisy User-generated Text (Brussels, Belgium), Association for Computational Linguistics, November 2018, pp. 164–172.

[8] Sergey Bartunov, Dmitry Kondrashkin, Anton Osokin, and Dmitry Vetrov, *Breaking sticks and ambiguities with adaptive skip-gram*, Proceedings of the 19th International Conference on Artificial Intelligence and Statistics (Cadiz, Spain) (Arthur Gretton and Christian C. Robert, eds.), Proceedings of Machine Learning Research, vol. 51, PMLR, 09–11 May 2016, pp. 130–138.

[9] Jacob Benesty, Jingdong Chen, Yiteng Huang, and Israel Cohen, *Pearson correlation coefficient*, Noise reduction in speech processing, Springer, 2009, pp. 1–4.

[10] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin, *A neural probabilistic language model*, Journal of machine learning research **3** (2003), no. Feb, 1137–1155.

[11] Tom Bosc and Pascal Vincent, *Auto-encoding dictionary definitions into consistent word embeddings*, Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (Brussels, Belgium), Association for Computational Linguistics, October-November 2018, pp. 1522–1532.

[12] Constantinos Boulis and Mari Ostendorf, *Text classification by augmenting the bag-of-words representation with redundancy-compensated bigrams*, Proc. of the International Workshop in Feature Selection in Data Mining, Citeseer, 2005, pp. 9–16.

[13] Eleftheria Briakou, Nikos Athanasiou, and Alexandros Potamianos, *Cross-topic distributional semantic representations via unsupervised mappings*, 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) (Minneapolis, Minnesota), Association for Computational Linguistics, June 2019, pp. 1052–1061.

[14] Jose Camacho-Collados and Mohammad Taher Pilehvar, *From word to sense embeddings: A survey on vector representations of meaning*, Journal of Artificial Intelligence Research **63** (2018), 743–788.

[15] Zixuan Cao, Yongmei Zhou, Aimin Yang, and Jiahui Fu, *Contextualized word representations with effective attention for aspect-based sentiment analysis*, Chinese Computational Linguistics (Cham) (Maosong Sun, Xuanjing Huang, Heng Ji, Zhiyuan Liu, and Yang Liu, eds.), Springer International Publishing, 2019, pp. 467–478.

[16] Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Brian Strope, and Ray Kurzweil, *Universal sentence encoder for English*, Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations (Brussels, Belgium), Association for Computational Linguistics, November 2018, pp. 169–174.

[17] Ting-Yun Chang and Yun-Nung Chen, *What does this word mean? explaining contextualized embeddings with natural language definition*, Proceedings of

the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP) (Hong Kong, China), Association for Computational Linguistics, November 2019, pp. 6064–6070.

[18] Ting-Yun Chang, Ta-Chung Chi, Shang-Chi Tsai, and Yun-Nung Chen, *xsense: Learning sense-separated sparse representations and textual definitions for explainable word sense networks*, arXiv preprint arXiv:1809.03348 (2018).

[19] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio, *Learning phrase representations using RNN encoder–decoder for statistical machine translation*, Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP) (Doha, Qatar), Association for Computational Linguistics, October 2014, pp. 1724–1734.

[20] Ronan Collobert and Jason Weston, *A unified architecture for natural language processing: Deep neural networks with multitask learning*, Proceedings of the 25th International Conference on Machine Learning (New York, NY, USA), ICML '08, Association for Computing Machinery, 2008, p. 160–167.

[21] Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes, *Supervised learning of universal sentence representations from natural language inference data*, Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (Copenhagen, Denmark), Association for Computational Linguistics, September 2017, pp. 670–680.

[22] Arpita Das, Harish Yenala, Manoj Chinnakotla, and Manish Shrivastava, *Together we stand: Siamese networks for similar question retrieval*, Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics

(Volume 1: Long Papers) (Berlin, Germany), Association for Computational Linguistics, August 2016, pp. 378–387.

[23] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova, *BERT: Pre-training of deep bidirectional transformers for language understanding*, 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) (Minneapolis, Minnesota), Association for Computational Linguistics, June 2019, pp. 4171–4186.

[24] Yuan Ding and Martha Palmer, *Machine translation using probabilistic synchronous dependency insertion grammars*, Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05) (Ann Arbor, Michigan), Association for Computational Linguistics, June 2005, pp. 541–548.

[25] Katrin Erk, Diana McCarthy, and Nicholas Gaylord, *Investigations on word senses and word usages*, Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1 - Volume 1 (USA), ACL '09, Association for Computational Linguistics, 2009, p. 10–18.

[26] Manaal Faruqui, Yulia Tsvetkov, Dani Yogatama, Chris Dyer, and Noah A. Smith, *Sparse overcomplete word vector representations*, 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) (Beijing, China), Association for Computational Linguistics, July 2015, pp. 1491–1500.

[27] Ana Fernández-Montraveta, Gloria Vázquez, and Christiane Fellbaum, *The spanish version of wordnet 3.0*, Text Resources and Lexical Knowledge. Mouton de Gruyter (2008), 175–182.

[28] Artyom Gadetsky, Ilya Yakubovskiy, and Dmitry Vetrov, *Conditional generators of words definitions*, Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers) (Melbourne, Australia), Association for Computational Linguistics, July 2018, pp. 266–271.

[29] Maria Giatsoglou, Manolis G. Vozalis, Konstantinos Diamantaras, Athena Vakali, George Sarigiannidis, and Konstantinos Ch. Chatzisavvas, *Sentiment analysis leveraging emotions and word embeddings*, Expert Systems with Applications **69** (2017), 214 – 224.

[30] Thomas L. Griffiths, Michael I. Jordan, Joshua B. Tenenbaum, and David M. Blei, *Hierarchical topic models and the nested chinese restaurant process*, Advances in Neural Information Processing Systems 16 (S. Thrun, L. K. Saul, and B. Schölkopf, eds.), MIT Press, 2004, pp. 17–24.

[31] Zellig S Harris, *Distributional structure*, Word **10** (1954), no. 2-3, 146–162.

[32] Michael A. Hedderich, Andrew Yates, Dietrich Klakow, and Gerard de Melo, *Using multi-sense vector embeddings for reverse dictionaries*, Proceedings of the 13th International Conference on Computational Semantics - Long Papers (Gothenburg, Sweden), Association for Computational Linguistics, May 2019, pp. 247–258.

[33] Felix Hill, Kyunghyun Cho, Anna Korhonen, and Yoshua Bengio, *Learning to understand phrases by embedding the dictionary*, Transactions of the Association for Computational Linguistics **4** (2016), 17–30.

[34] Eric H. Huang, Richard Socher, Christopher D. Manning, and Andrew Y. Ng, *Improving word representations via global context and multiple word prototypes*, Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1 (USA), ACL '12, Association for Computational Linguistics, 2012, p. 873–882.

[35] John Hutchins, *The history of machine translation in a nutshell*, Retrieved from `http://www.hutchinsweb.me.uk/Nutshell-2005.pdf` (2005).

[36] Hitoshi Isahara, Francis Bond, Kiyotaka Uchimoto, Masao Utiyama, and Kyoko Kanzaki, *Development of the Japanese WordNet*, Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08) (Marrakech, Morocco), European Language Resources Association (ELRA), May 2008.

[37] Shonosuke Ishiwatari, Hiroaki Hayashi, Naoki Yoshinaga, Graham Neubig, Shoetsu Sato, Masashi Toyoda, and Masaru Kitsuregawa, *Learning to describe phrases with local and global contexts*, arXiv preprint arXiv:1811.00266 (2018).

[38] Eric Jang, Shixiang Gu, and Ben Poole, *Categorical reparameterization with gumbel-softmax*, arXiv preprint arXiv:1611.01144 (2016).

[39] Sujay Kumar Jauhar, Chris Dyer, and Eduard Hovy, *Ontologically grounded multi-sense representation learning for semantic vector space models*, Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Denver, Colorado), Association for Computational Linguistics, May–June 2015, pp. 683–693.

[40] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov, *Bag of tricks for efficient text classification*, Proceedings of the 15th Conference of the

European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers (Valencia, Spain), Association for Computational Linguistics, April 2017, pp. 427–431.

[41] Scharolta Katharina Sienčnik, *Adapting word2vec to named entity recognition*, Proceedings of the 20th Nordic Conference of Computational Linguistics (NODALIDA 2015) (Vilnius, Lithuania), Linköping University Electronic Press, Sweden, May 2015, pp. 239–243.

[42] keitakurita, *Paper dissected: "attention is all you need" explained*, `http://mlexplained.com/2017/12/29/attention-is-all-you-need-explained/`, 2017, [Online; accessed 2-June-2020].

[43] Tom Kenter, Alexey Borisov, and Maarten de Rijke, *Siamese CBOW: Optimizing word embeddings for sentence representations*, Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (Berlin, Germany), Association for Computational Linguistics, August 2016, pp. 941–951.

[44] Yoon Kim, Yacine Jernite, David Sontag, and Alexander M. Rush, *Character-aware neural language models*, Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, AAAI'16, AAAI Press, 2016, p. 2741–2749.

[45] Tom Kocmi and Ondřej Bojar, *An exploration of word embedding initialization in deep-learning tasks*, Proceedings of the 14th International Conference on Natural Language Processing (ICON-2017) (Kolkata, India), NLP Association of India, December 2017, pp. 56–64.

[46] Philipp Koehn, *Europarl: A parallel corpus for statistical machine translation*, MT summit, vol. 5, Citeseer, 2005, pp. 79–86.

[47] Quoc Le and Tomas Mikolov, *Distributed representations of sentences and documents*, Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32, ICML'14, JMLR.org, 2014, p. II–1188–II–1196.

[48] Guang-He Lee and Yun-Nung Chen, *MUSE: Modularizing unsupervised sense embeddings*, Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (Copenhagen, Denmark), Association for Computational Linguistics, September 2017, pp. 327–337.

[49] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer, *Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension*, arXiv preprint arXiv:1910.13461 (2019).

[50] Jiwei Li and Dan Jurafsky, *Do multi-sense embeddings improve natural language understanding?*, Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (Lisbon, Portugal), Association for Computational Linguistics, September 2015, pp. 1722–1732.

[51] Yijia Liu, Wanxiang Che, Yuxuan Wang, Bo Zheng, Bing Qin, and Ting Liu, *Deep contextualized word embeddings for universal dependency parsing*, ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP) **19** (2019), no. 1, 1–17.

[52] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov, *Roberta: A robustly optimized bert pretraining approach*, arXiv preprint arXiv:1907.11692 (2019).

[53] J. Lu, C. Xiong, D. Parikh, and R. Socher, *Knowing when to look: Adaptive attention via a visual sentinel for image captioning*, 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (Los Alamitos, CA, USA), IEEE Computer Society, jul 2017, pp. 3242–3250.

[54] Minh-Thang Luong, Quoc V Le, Ilya Sutskever, Oriol Vinyals, and Lukasz Kaiser, *Multi-task sequence to sequence learning*, arXiv preprint arXiv:1511.06114 (2015).

[55] Sean MacAvaney, Andrew Yates, Arman Cohan, and Nazli Goharian, *Cedr: Contextualized embeddings for document ranking*, Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (New York, NY, USA), SIGIR'19, Association for Computing Machinery, 2019, p. 1101–1104.

[56] Justin Christopher Martineau and Tim Finin, *Delta tfidf: An improved feature space for sentiment analysis*, Third international AAAI conference on weblogs and social media (San Jose, USA), 2009.

[57] Diana McCarthy, Marianna Apidianaki, and Katrin Erk, *Word sense clustering and clusterability*, Computational Linguistics **42** (2016), no. 2, 245–275.

[58] Marek Medved and Ales Horák, *Sentence and word embedding employed in open question-answering.*, ICAART (2), 2018, pp. 486–492.

[59] Timothee Mickus, Denis Paperno, and Matthieu Constant, *Mark my word: A sequence-to-sequence approach to definition modeling*, Proceedings of the First NLPL Workshop on Deep Learning for Natural Language Processing (Turku, Finland), Linköping University Electronic Press, September 2019, pp. 1–11.

[60] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean, *Efficient estimation of word representations in vector space*, arXiv preprint arXiv:1301.3781 (2013).

[61] Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur, *Recurrent neural network based language model*, Eleventh annual conference of the international speech communication association (Florence, Italy), 2010.

[62] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean, *Distributed representations of words and phrases and their compositionality*, Advances in Neural Information Processing Systems 26 (C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, eds.), Curran Associates, Inc., 2013, pp. 3111–3119.

[63] George A Miller, *Wordnet: An electronic lexical database*, MIT press, 1998.

[64] Leann Myers and Maria J Sirois, *Spearman correlation coefficients, differences between*, Encyclopedia of statistical sciences **12** (2004).

[65] Roberto Navigli and Simone Paolo Ponzetto, *Babelnet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network*, Artif. Intell. **193** (2012), 217–250.

[66] Arvind Neelakantan, Jeevan Shankar, Alexandre Passos, and Andrew McCallum, *Efficient non-parametric estimation of multiple embeddings per word in vector space*, Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP) (Doha, Qatar), Association for Computational Linguistics, October 2014, pp. 1059–1069.

[67] Dai Quoc Nguyen, Dat Quoc Nguyen, Ashutosh Modi, Stefan Thater, and Manfred Pinkal, *A mixture model for learning multi-sense word embeddings*,

Proceedings of the 6th Joint Conference on Lexical and Computational Semantics (*SEM 2017) (Vancouver, Canada), Association for Computational Linguistics, August 2017, pp. 121–127.

[68] Ke Ni and William Yang Wang, *Learning to explain non-standard English words and phrases*, Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers) (Taipei, Taiwan), Asian Federation of Natural Language Processing, November 2017, pp. 413–417.

[69] Thanapon Noraset, Chen Liang, Larry Birnbaum, and Doug Downey, *Definition modeling: Learning to define word embeddings in natural language*, Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, AAAI'17, AAAI Press, 2017, p. 3259–3266.

[70] Matteo Pagliardini, Prakhar Gupta, and Martin Jaggi, *Unsupervised learning of sentence embeddings using compositional n-gram features*, Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers) (New Orleans, Louisiana), Association for Computational Linguistics, June 2018, pp. 528–540.

[71] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu, *Bleu: A method for automatic evaluation of machine translation*, Proceedings of the 40th Annual Meeting on Association for Computational Linguistics (USA), ACL '02, Association for Computational Linguistics, 2002, p. 311–318.

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

[73] Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer, *Deep contextualized word representations*, Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers) (New Orleans, Louisiana), Association for Computational Linguistics, June 2018, pp. 2227–2237.

[74] Mohammad Taher Pilehvar and Nigel Collier, *De-conflated semantic representations*, Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (Austin, Texas), Association for Computational Linguistics, November 2016, pp. 1680–1690.

[75] Andreas Poyiatzis, *Nlp: Contextualized word embeddings from bert*, https://towardsdatascience.com/nlp-extract-contextualized-word-embeddings-from-bert-keras-tf-67ef29f60a7b, 2019, [Online; accessed 3-June-2020].

[76] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever, *Improving language understanding by generative pre-training*, https://www.cs.ubc.ca/ amuham01/LING530/papers/radford2018improving.pdf, 2018, [Online; accessed 2-June-2020].

[77] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever, *Language models are unsupervised multitask learners*, OpenAI Blog **1** (2019), no. 8, 9.

[78] Nils Reimers and Iryna Gurevych, *Sentence-BERT: Sentence embeddings using Siamese BERT-networks*, Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP) (Hong Kong,

China), Association for Computational Linguistics, November 2019, pp. 3982–3992.

[79] Joseph Reisinger and Raymond J. Mooney, *Multi-prototype vector-space models of word meaning*, Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics (Los Angeles, California), Association for Computational Linguistics, June 2010, pp. 109–117.

[80] David Rozado, *Wide range screening of algorithmic bias in word embedding models using large sentiment lexicons reveals underreported bias types*, PLOS ONE **15** (2020), no. 4, 1–26.

[81] Herbert Rubenstein and John B Goodenough, *Contextual correlates of synonymy*, Communications of the ACM **8** (1965), no. 10, 627–633.

[82] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf, *Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter*, arXiv preprint arXiv:1910.01108 (2019).

[83] Richard Socher, Cliff Chiung-Yu Lin, Andrew Y. Ng, and Christopher D. Manning, *Parsing natural scenes and natural language with recursive neural networks*, Proceedings of the 28th International Conference on International Conference on Machine Learning (Madison, WI, USA), ICML'11, Omnipress, 2011, p. 129–136.

[84] Sofia Stamou, Goran Nenadic, and Dimitris Christodoulakis, *Exploring balkanet shared ontology for multilingual conceptual indexing*, Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04) (Lisbon, Portugal), European Language Resources Association (ELRA), May 2004.

[85] Anant Subramanian, Danish Pruthi, Harsh Jhamtani, Taylor Berg-Kirkpatrick, and Eduard Hovy, *Spine: Sparse interpretable neural embeddings*, Thirty-Second AAAI Conference on Artificial Intelligence (Hilton New Orleans Riverside, USA), 2018.

[86] Fei Tian, Hanjun Dai, Jiang Bian, Bin Gao, Rui Zhang, Enhong Chen, and Tie-Yan Liu, *A probabilistic model for learning multi-prototype word embeddings*, Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers (Dublin, Ireland), Dublin City University and Association for Computational Linguistics, August 2014, pp. 151–160.

[87] Joseph Turian, Lev-Arie Ratinov, and Yoshua Bengio, *Word representations: A simple and general method for semi-supervised learning*, Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (Uppsala, Sweden), Association for Computational Linguistics, July 2010, pp. 384–394.

[88] AM Turing, *Computing machinery and intelligence*, Mind **59** (1950), no. 236, 433–460.

[89] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin, *Attention is all you need*, Advances in Neural Information Processing Systems 30 (I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds.), Curran Associates, Inc., 2017, pp. 5998–6008.

[90] John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu, *Towards universal paraphrastic sentence embeddings*, arXiv preprint arXiv:1511.08198 (2015).

[91] Shuntaro Yada, *A review of deep contextualized word representations*, https://www.slideshare.net/shuntaroy/a-review-of-deep-contextualized-word-representations-peters-2018, 2018, [Online; accessed 2-June-2020].

[92] Dongqiang Yang and David M. W. Powers, *Measuring semantic similarity in the taxonomy of wordnet*, Proceedings of the Twenty-Eighth Australasian Conference on Computer Science - Volume 38 (AUS), ACSC '05, Australian Computer Society, Inc., 2005, p. 315–322.

[93] L. Yang, C. Kong, Y. Chen, Y. Liu, Q. Fan, and E. Yang, *Incorporating sememes into chinese definition modeling*, IEEE/ACM Transactions on Audio, Speech, and Language Processing **28** (2020), 1669–1677.

[94] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le, *Xlnet: Generalized autoregressive pretraining for language understanding*, Advances in neural information processing systems, 2019, pp. 5754–5764.

[95] Chaonan Ye and Wanling Liu, *Squad 2.0 question answering system*, https://web.stanford.edu/class/archive/cs/cs224n/cs224n.1194/posters/15846591.pdf, [Online; accessed 20-June-2020].

[96] JS Yu, SW Yu, Yang Liu, and Huarui Zhang, *Introduction to chinese concept dictionary*, International Conference on Chinese Computing (ICCC2001), 2001, pp. 361–367.

[97] Haitong Zhang, Yongping Du, Jiaxin Sun, and Qingxiao Li, *Improving interpretability of word embeddings by generating definition and usage*, Expert Systems with Applications (2020), 113633.

[98] Zhendong Dong and Qiang Dong, *Hownet - a hybrid language and knowledge resource*, International Conference on Natural Language Processing and Knowledge Engineering, 2003. Proceedings. 2003, 2003, pp. 820–824.

[99] Ruimin Zhu, Thanapon Noraset, Alisa Liu, Wenxin Jiang, and Doug Downey, *Multi-sense definition modeling using word sense decompositions*, arXiv preprint arXiv:1909.09483 (2019).

[100] Will Y Zou, Richard Socher, Daniel Cer, and Christopher D Manning, *Bilingual word embeddings for phrase-based machine translation*, Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, 2013, pp. 1393–1398.

Vita

Candidate's full name: Arman Kabiri

University attended (with dates and degrees obtained):
Shahrekord University, Iran, Bachelor of Software Engineering, 2013 – 2017

Publications:
Arman Kabiri, Paul Cook, *Evaluating a Multi-sense Definition Generation Model for Multiple Languages*, to be orally presented in 23rd International Conference on Text, Speech and Dialogue (TSD 2020), 2020