Mark my Word: A Sequence-to-Sequence Approach to Definition Modeling

Timothee Mickus, Denis Paperno, Mathieu Constant

Mark my Word:
A Sequence-to-Sequence Approach to Definition Modeling

Timothee Mickus
Université de Lorraine
CNRS, ATILF
tmickus@atilf.fr

Denis Paperno
Utrecht University
d.paperno@uu.nl

Mathieu Constant
Université de Lorraine
CNRS, ATILF
mconstant@atilf.fr

Abstract

Defining words in a textual context is a useful task both for practical purposes and for gaining insight into distributed word representations. Building on the distributional hypothesis, we argue here that the most natural formalization of definition modeling is to treat it as a sequence-to-sequence task, rather than a word-to-sequence task: given an input sequence with a highlighted word, generate a contextually appropriate definition for it. We implement this approach in a Transformer-based sequence-to-sequence model. Our proposal allows to train contextualization and definition generation in an end-to-end fashion, which is a conceptual improvement over earlier works. We achieve state-of-the-art results both in contextual and non-contextual definition modeling.

1 Introduction

The task of *definition modeling*, introduced by Noraset et al. (2017), consists in generating the dictionary definition of a specific word: for instance, given the word "*monotreme*" as input, the system would need to produce a definition such as "*any of an order (Monotremata) of egg-laying mammals comprising the platypuses and echidnas*".[1] Following the tradition set by lexicographers, we call the word being defined a *definiendum* (pl. *definienda*), whereas a word occurring in its definition is called a *definiens* (pl. *definientia*).

Definition modeling can prove useful in a variety of applications. Systems trained for the task may generate dictionaries for low resource languages, or extend the coverage of existing lexicographic resources where needed, e.g. of domain-specific vocabulary. Such systems may also be able to provide reading help by giving definitions for words in the text.

A major intended application of definition modeling is the explication and evaluation of distributed lexical representations, also known as word embeddings (Noraset et al., 2017). This evaluation procedure is based on the postulate that the meaning of a word, as is captured by its embedding, should be convertible into a human-readable dictionary definition. How well the meaning is captured must impact the ability of the model to reproduce the definition, and therefore embedding architectures can be compared according to their downstream performance on definition modeling. This intended usage motivates the requirement that definition modeling architectures take as input the embedding of the *definiendum* and not retrain it.

From a theoretical point of view, usage of word embeddings as representations of meaning (cf. Lenci, 2018; Boleda, 2019, for an overview) is motivated by the distributional hypothesis (Harris, 1954). This framework holds that meaning can be inferred from the linguistic context of the word, usually seen as co-occurrence data. The context of usage is even more crucial for characterizing meanings of ambiguous or polysemous words: a definition that does not take disambiguating context into account will be of limited use (Gadetsky et al., 2018).

We argue that definition modeling should preserve the link between the *definiendum* and its context of occurrence. The most natural approach to this task is to treat it as a *sequence-to-sequence* task, rather than a *word-to-sequence* task: given an input sequence with a highlighted word, generate a contextually appropriate definition for it (cf. sections 3 & 4). We implement this approach in a Transformer-based sequence-to-sequence model that achieves state-of-the-art performances (sections 5 & 6).

[1] Definition from Merriam-Webster.

Proceedings of the First NLPL Workshop on Deep Learning for Natural Language Processing, pages 1–11

2 Related Work

In their seminal work on definition modeling, Noraset et al. (2017) likened systems generating definitions to language models, which can naturally be used to generate arbitrary text. They built a sequential LSTM seeded with the embedding of the *definiendum*; its output at each time-step was mixed through a gating mechanism with a feature vector derived from the *definiendum*.

Gadetsky et al. (2018) stressed that a *definiendum* outside of its specific usage context is ambiguous between all of its possible definitions. They proposed to first compute the AdaGram vector (Bartunov et al., 2016) for the *definiendum*, to then disambiguate it using a gating mechanism learned over contextual information, and finally to run a language model over the sequence of *definientia* embeddings prepended with the disambiguated *definiendum* embedding.

In an attempt to produce a more interpretable model, Chang et al. (2018) map the *definiendum* to a sparse vector representation. Their architecture comprises four modules. The first encodes the context in a sentence embedding, the second converts the *definiendum* into a sparse vector, the third combines the context embedding and the sparse representation, passing them on to the last module which generates the definition.

Related to these works, Yang et al. (2019) specifically tackle definition modeling in the context of Chinese—whereas all previous works on definition modeling studied English. In a Transformer-based architecture, they incorporate "sememes" as part of the representation of the *definiendum* to generate definitions.

On a more abstract level, definition modeling is related to research on the analysis and evaluation of word embeddings (Levy and Goldberg, 2014a,b; Arora et al., 2018; Batchkarov et al., 2016; Swinger et al., 2018, e.g.). It also relates to other works associating definitions and embeddings, like the "reverse dictionary task" (Hill et al., 2016)—retrieving the *definiendum* knowing its definition, which can be argued to be the opposite of definition modeling—or works that derive embeddings from definitions (Wang et al., 2015; Tissier et al., 2017; Bosc and Vincent, 2018).

3 Definition modeling as a sequence-to-sequence task

Gadetsky et al. (2018) remarked that words are often ambiguous or polysemous, and thus generating a correct definition requires that we either use sense-level representations, or that we disambiguate the word embedding of the *definiendum*. The disambiguation that Gadetsky et al. (2018) proposed was based on a contextual cue—ie. a short text fragment. As Chang et al. (2018) notes, the cues in Gadetsky et al.'s (2018) dataset do not necessarily contain the *definiendum* or even an inflected variant thereof. For instance, one training example disambiguated the word "*fool*" using the cue "*enough horsing around—let's get back to work!*".

Though the remark that *definienda* must be disambiguated is pertinent, the more natural formulation of such a setup would be to disambiguate the *definiendum* using its actual context of occurrence. In that respect, the *definiendum* and the contextual cue would form a linguistically coherent sequence, and thus it would make sense to encode the context together with the *definiendum*, rather than to merely rectify the *definiendum* embedding using a contextual cue. Therefore, definition modeling is by its nature a sequence-to-sequence task: mapping contexts of occurrence of *definienda* to definitions.

This remark can be linked to the distributional hypothesis (Harris, 1954). The distributional hypothesis suggests that a word's meaning can be inferred from its context of usage; or, more succinctly, that "you shall know a word by the company it keeps" (Firth, 1957). When applied to definition modeling, the hypothesis can be rephrased as follows: the correct definition of a word can only be given when knowing in what linguistic context(s) it occurs. Though different kinds of linguistic contexts have been suggested throughout the literature, we remark here that sentential context may sometimes suffice to guess the meaning of a word that we don't know (Lazaridou et al., 2017). Quoting from the example above, the context "*enough* _____ *around—let's get back to work!*" sufficiently characterizes the meaning of the omitted verb to allow for an approximate definition for it even if the blank is not filled (Taylor, 1953; Devlin et al., 2018).

This reformulation can appear contrary to the original proposal by Noraset et al. (2017), which conceived definition modeling as a "word-to-sequence task". They argued for an approach related to, though distinct from sequence-to-

sequence architectures. Concretely, a specific encoding procedure was applied to the *definiendum*, so that it could be used as a feature vector during generation. In the simplest case, vector encoding of the *definiendum* consists in looking up its vector in a vocabulary embedding matrix.

We argue that the whole context of a word's usage should be accessible to the generation algorithm rather than a single vector. To take a more specific case of verb definitions, we observe that context explicitly represents argument structure, which is obviously useful when defining the verb. There is no guarantee that a single embedding, even if it be contextualized, would preserve this wealth of information—that is to say, that you can cram all the information pertaining to the syntactic context into a single vector.

Despite some key differences, all of the previously proposed architectures we are aware of (Noraset et al., 2017; Gadetsky et al., 2018; Chang et al., 2018; Yang et al., 2019) followed a pattern similar to sequence-to-sequence models. They all implicitly or explicitly used distinct submodules to encode the *definiendum* and to generate the *definientia*. In the case of Noraset et al. (2017), the encoding was the concatenation of the embedding of the *definiendum*, a vector representation of its sequence of characters derived from a character-level CNN, and its "hypernym embedding". Gadetsky et al. (2018) used a sigmoid-based gating module to tweak the *definiendum* embedding. The architecture proposed by Chang et al. (2018) is comprised of four modules, only one of which is used as a decoder: the remaining three are meant to convert the *definiendum* as a sparse embedding, select some of the sparse components of its meaning based on a provided context, and encode it into a representation adequate for the decoder.

Aside from theoretical implications, there is another clear gain in considering definition modeling as a sequence-to-sequence task. Recent advances in embedding designs have introduced contextual embeddings (McCann et al., 2017; Peters et al., 2018; Devlin et al., 2018); and these share the particularity that they are a "function of the entire sentence" (Peters et al., 2018): in other words, vector representations are assigned to tokens rather than to word types, and moreover semantic information about a token can be distributed over other token representations. To extend definition modeling to contextual embeddings therefore requires that we devise architectures able to encode a word in its context; in that respect sequence-to-sequence architectures are a natural choice.

A related point is that not all *definienda* are comprised of a single word: multi-word expressions include multiple tokens, yet receive a single definition. Word embedding architectures generally require a pre-processing step to detect these expressions and merge them into a single token. However, as they come with varying degrees of semantic opacity (Cordeiro et al., 2016), a definition modeling system would benefit from directly accessing the tokens they are made up from. Therefore, if we are to address the entirety of the language and the entirety of existing embedding architectures in future studies, reformulating definition modeling as a sequence-to-sequence task becomes a necessity.

4 Formalization

A sequence-to-sequence formulation of definition modeling can formally be seen as a mapping between contexts of occurrence of *definienda* and their corresponding definitions. It moreover requires that the *definiendum* be formally distinguished from the remaining context: otherwise the definition could not be linked to any particular word of the contextual sequence, and thus would need to be equally valid for any word of the contextual sequence.

We formalize definition modeling as mapping to sequences of *definientia* from sequences of pairs $\langle w_1, i_1 \rangle, \ \ldots, \ \langle w_n, i_n \rangle$, where w_k is the k^{th} word in the input and $i_k \in \{0, 1\}$ indicates whether the k^{th} token is to be defined. As only one element of the sequence should be highlighted, we expect the set of all indicators to contain only two elements: the one, $i_d = 1$, to mark the *definiendum*, the other, $i_c = 0$, to mark the context; this entails that we encode this marking using one bit only.[2]

To treat definition modeling as a sequence-to-sequence task, the information from each pair $\langle w_k, i_k \rangle$ has to be integrated into a single repre-

[2]Multiple instances of the same *definiendum* within a single context should all share a single definition, and therefore could theoretically all be marked using the *definiendum* indicator $i_d = 1$. Likewise the words that make up a multi-word expression should all be marked with this i_d indicator. In this work, however, we only mark a single item; in cases when multiple occurrences of the same *definiendum* were attested, we simply marked the first occurrence.

sentation $\vec{marked}_k$:

$$\vec{marked}_k = mark(i_k, \vec{w}_k) \quad (1)$$

This marking function can theoretically take any form. Considering that definition modeling uses the embedding of the definiendum $\vec{w}_d = e(w_d)$, in this work we study a multiplicative and an additive mechanism, as they are conceptually the simplest form this marking can take in a vector space. They are given schematically in Figure 1, and formally defined as:

$$\vec{marked}_k^{\times} = i_k \times \vec{w}_k \quad (2)$$

$$\vec{marked}_k^{+} = e(i_k) + \vec{w}_k \quad (3)$$

The last point to take into account is where to set the marking. Two natural choices are to set it either before or after encoded representations were obtained. We can formalize this using either of the following equation, with $\mathcal{E}$ the model's encoder:

$$\vec{marked}_k^{\text{after}} = mark(i_k, \mathcal{E}(\vec{w}_k))$$
$$\vec{marked}_k^{\text{before}} = \mathcal{E}(mark(i_k, \vec{w}_k)) \quad (4)$$

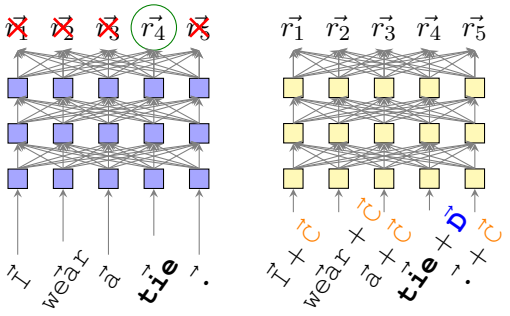
4.1 Multiplicative marking: SELECT

The first option we consider is to use scalar multiplication to distinguish the word to define. In such a scenario, the marked token encoding is

$$\vec{marked}_k^{\times} = i_k \times \vec{w}_k \quad (2)$$

As we use bit information as indicators, this form of marking entails that only the representation of the *definiendum* be preserved and that all other contextual representations are set to $\vec{0} = (0, \cdots, 0)$: thus multiplicative marking amounts to selecting just the *definiendum* embedding and discarding other token embeddings. The contextualized *definiendum* encoding bears the trace of its context, but detailed information is irreparably lost. Hence, we refer to such an integration mechanism as a SELECT marking of the *definiendum*.

When to apply marking, as introduced by eq. 4, is crucial when using the multiplicative marking scheme SELECT. Should we mark the *definiendum* before encoding, then only the *definiendum* embedding is passed into the encoder: the resulting system provides out-of-context definitions, like in Noraset et al. (2017) where the definition is not linked to the context of a word but to its *definiendum* only. For context to be taken into account



(a) SELECT: Selecting from encoded items. Items are contextualized and the *definiendum* is singled out from them.

(b) ADD: Additive marking in encoder. Context items and *definiendum* are marked by adding dedicated embeddings.

Figure 1: Additive vs. multiplicative integration

under the multiplicative strategy, tokens w_k must be encoded and contextualized before integration with the indicator i_k.

Figure 1a presents the contextual SELECT mechanism visually. It consists in coercing the decoder to attend only to the contextualized representation for the *definiendum*. To do so, we encode the full context and then select only the encoded representation of the *definiendum*, dropping the rest of the context, before running the decoder. In the case of the Transformer architecture, this is equivalent to using a multiplicative marking on the encoded representations: vectors that have been zeroed out are ignored during attention and thus cannot influence the behavior of the decoder.

This SELECT approach may seem intuitive and naturally interpretable, as it directly controls what information is passed to the decoder—we carefully select only the contextualized *definiendum*, thus the only remaining zone of uncertainty would be how exactly contextualization is performed. It also seems to provide a strong and reasonable bias for training the definition generation system. Such an approach, however, is not guaranteed to excel: forcibly omitted context could contain important information that might not be easily incorporated in the *definiendum* embedding.

Being simple and natural, the SELECT approach resembles architectures like that of Gadetsky et al. (2018) and Chang et al. (2018): the full encoder is dedicated to altering the embedding of the *definiendum* on the basis of its context; in that, the encoder may be seen as a dedicated contextualization sub-module.

4.2 Additive marking: ADD

We also study an additive mechanism shown in Figure 1b (henceforth ADD). It concretely consists in embedding the word w_k and its indicator bit i_k in the same vector space and adding the corresponding vectors:

$$\vec{marked}_k^+ = e(i_k) + \vec{w}_k \quad (3)$$

In other words, under ADD we distinguish the *definiendum* by adding a vector $\vec{\mathtt{D}}$ to the *definiendum* embedding, and another vector $\vec{\mathtt{C}}$ to the remaining context token embeddings; both markers $\vec{\mathtt{D}}$ and $\vec{\mathtt{C}}$ are learned during training. In our implementation, markers are added to the input of the encoder, so that the encoder has access to this information; we leave the question of whether to integrate indicators and words at other points of the encoding process, as suggested in eq. 4, to future work.

Additive marking of substantive features has its precedents. For example, BERT embeddings (Devlin et al., 2018) are trained using two sentences at once as input; sentences are distinguished with added markers called "segment encodings". Tokens from the first sentence are all marked with an added vector $\vec{\mathtt{segment}}_A$, whereas tokens from second sentences are all marked with an added vector $\vec{\mathtt{segment}}_B$. The main difference here is that we only mark one item with the marker $\vec{D}$, while all others are marked with $\vec{C}$.

This ADD marking is more expressive than the SELECT architecture. Sequence-to-sequence decoders typically employ an attention to the input source (Bahdanau et al., 2014), which corresponds to a re-weighting of the encoded input sequence based on a similarity between the current state of the decoder (the 'query') and each member of the input sequence (the 'keys'). This re-weighting is normalized with a softmax function, producing a probability distribution over keys. However, both non-contextual definition modeling and the SELECT approach produce singleton encoded sequences: in such scenarios the attention mechanism assigns a single weight of 1 and thus devolves into a simple linear transformation of the value and makes the attention mechanism useless. Using an additive marker, rather than a selective mechanism, will prevent this behavior.

5 Evaluation

We implement several sequence to sequence models with the Transformer architecture (Vaswani et al., 2017), building on the OpenNMT library (Klein et al., 2017) with adaptations and modifications when necessary.[3] Throughout this work, we use GloVe vectors (Pennington et al., 2014) and freeze weights of all embeddings for a fairer comparison with previous models; words not in GloVe but observed in train or validation data and missing *definienda* in our test sets were randomly initialized with components drawn from a normal distribution $\mathcal{N}(0, 1)$.

We train a distinct model for each dataset. We batch examples by 8,192, using gradient accumulation to circumvent GPU limitations. We optimize the network using Adam with $\beta_1 = 0.99$, $\beta_2 = 0.998$, a learning rate of 2, label smoothing of 0.1, Noam exponential decay with 2000 warmup steps, and dropout rate of 0.4. The parameters are initialized using Xavier. Models were trained for up to 120,000 steps with checkpoints at each 1000 steps; we stopped training if perplexity on the validation dataset stopped improving. We report results from checkpoints performing best on validation.

5.1 Implementation of the Non-contextual Definition Modeling System

In non-contextual definition modeling, *definienda* are mapped directly to definitions. As the source corresponds only to the *definiendum*, we conjecture that few parameters are required for the encoder. We use 1 layer for the encoder, 6 for the decoder, 300 dimensions per hidden representations and 6 heads for multi-head attention. We do not share vocabularies between the encoder and the decoder: therefore output tokens can only correspond to words attested as *definientia*.[4]

The dropout rate and warmup steps number were set using a hyperparameter search on the dataset from Noraset et al. (2017), during which encoder and decoder vocabulary were merged for computational simplicity and models stopped after 12,000 steps. We first fixed dropout to 0.1 and tested warmup step values between 1000 and

[3] Code & data are available at the following URL: https://github.com/TimotheeMickus/onmt-selectrans.

[4] In our case, not sharing vocabularies prevents the model from considering rare words only used as *definienda*, such as "*penumbra*" as potential outputs, and was found to improve performances.

10,000 by increments of 1000, then focused on the most promising span (1000–4000 steps) and exhaustively tested dropout rates from 0.2 to 0.8 by increments of 0.1.

5.2 Implementation of Contextualized Definition Modeling Systems

To compare the effects of the two integration strategies that we discussed in section 4, we implement both the additive marking approach (ADD, cf. section 4.2) and the alternative 'encode and select' approach (SELECT, cf. section 4.1). To match with the complex input source, we define encoders with 6 layers; we reemploy the set of hyperparameters previously found for the non-contextual system. Other implementation details, initialization strategies and optimization algorithms are kept the same as described above for the non-contextual version of the model.

We stress that the two approaches we compare for contextualizing the definiendum are applicable to almost any sequence-to-sequence neural architecture with an attention mechanism to the input source.[5] Here we chose to rely on a Transformer-based architecture (Vaswani et al., 2017), which has set the state of the art in a wide range of tasks, from language modeling (Dai et al., 2019) to machine translation (Ott et al., 2018). It is therefore expected that the Transformer architecture will also improve performances for definition modeling, if our arguments for treating it as a sequence to sequence task are on the right track.

5.3 Datasets

We train our models on three distinct datasets, which are all borrowed or adapted from previous works on definition modeling. As a consequence, our experiments focus on the English language. The dataset of Noraset et al. (2017) (henceforth $D_{\textbf{Nor}}$) maps *definienda* to their respective *definientia*, as well as additional information not used here. In the dataset of Gadetsky et al. (2018) (henceforth $D_{\textbf{Gad}}$), each example consists of a *definiendum*, the *definientia* for one of its meanings and a contextual cue sentence. D_{Nor} contains on average shorter definitions than D_{Gad}. Definitions in D_{Nor} have a mean length of 6.6 and a standard deviation of 5.78, whereas those in D_{Gad} have a mean length of 11.01 and a standard deviation of 6.96.

[5] For best results, the SELECT mechanism should require a bi-directional encoding mechanism.

Chang et al. (2018) stress that the dataset D_{Gad} includes many examples where the *definiendum* is absent from the associated cue. About half of these cues doe not contain an exact match for the corresponding *definiendum*, but up to 80% contains either an exact match or an inflected form of the *definiendum* according to lemmatization by the NLTK toolkit (Loper and Bird, 2002). To cope with this problematic characteristic, we converted the dataset into the word-in-context format assumed by our model by concatenating the *definiendum* with the cue. To illustrate this, consider the actual input from D_{Gad} comprised of the *definiendum* "*fool*" and its associated cue "*enough horsing around—let's get back to work!*": to convert this into a single sequence, we simply prepend the *definiendum* to the cue, which results in the sequence "*fool enough horsing around—let's get back to work!*". Hence the input sequences of D_{Gad} do not constitute linguistically coherent sequences, but it does guarantee that our sequence-to-sequence variants have access to the same input as previous models; therefore the inclusion of this dataset in our experiments is intended mainly for comparison with previous architectures. We also note that this conversion procedure entails that our examples have a very regular structure: the word marked as a *definiendum* is always the first word in the input sequence.

Our second strategy was to restrict the dataset by selecting only cues where the *definiendum* (or its inflected form) is present. The curated dataset (henceforth $D_{\textbf{Ctx}}$) contains 78,717 training examples, 9,413 for validation and 9,812 for testing. In each example, the first occurrence of the *definiendum* is annotated as such. D_{Ctx} thus differs from D_{Gad} in two ways: some definitions have been removed, and the exact citation forms of the *definienda* are not given. Models trained on D_{Ctx} implicitly need to lemmatize the *definiendum*, since inflected variants of a given word are to be aligned to a common representation; thus they are not directly comparable with models trained with the citation form of the *definiendum* that solely use context as a cue—viz. Gadetsky et al. (2018) & Chang et al. (2018). All this makes D_{Ctx} harder, but at the same time closer to a realistic application than the other two datasets, since each word appears inflected and in a specific *sentential context*. For applications of definition modeling, it would only be beneficial to

take up these challenges; for example, the output "*monotremes: plural of monotreme*"[6] would not have been self-contained, necessitating a second query for "*monotreme*".

5.4 Results

We use perplexity, a standard metric in definition modeling, to evaluate and compare our models. Informally, perplexity assesses the model's confidence in producing the ground-truth output when presented the source input. It is formally defined as the exponentiation of cross-entropy. We do not report BLEU or ROUGE scores due to the fact that an important number of ground-truth definitions are comprised of a single word, in particular in D_{Nor} ($\approx 25\%$). Single word outputs can either be assessed as entirely correct or entirely wrong using BLEU or ROUGE. However consider for instance the word "*elation*": that it be defined either as "*mirth*" or "*joy*" should only influence our metric slightly, and not be discounted as a completely wrong prediction.

	D_{Nor}	D_{Gad}	D_{Ctx}
Noraset et al.	48.168	45.620	–
Gadetsky et al.	–	43.540	–
Non-contextual	42.199	39.428	48.266
ADD	–	33.678	43.695
SELECT	–	33.998	62.039

Table 1: Results (perplexity)

Table 1 describes our main results in terms of perplexity. We do not compare with Chang et al. (2018), as they did not report the perplexity of their system and focused on a different dataset; likewise, Yang et al. (2019) consider only the Chinese variant of the task. Perplexity measures for Noraset et al. (2017) and Gadetsky et al. (2018) are taken from the authors' respective publications.

All our models perform better than previous proposals, by a margin of 4 to 10 points, for a relative improvement of 11–23%. Part of this improvement may be due to our use of Transformer-based architectures (Vaswani et al., 2017), which is known to perform well on semantic tasks (Radford, 2018; Cer et al., 2018; Devlin et al., 2018; Radford et al., 2019, eg.). Like Gadetsky et al. (2018), we conclude that disambiguating the *definiendum*, when done correctly, improves performances: our best performing contextual model outranks the non-contextual variant by 5 to 6 points. The marking of the definiendum out of its context (ADD vs. SELECT) also impacts results. Note also that we do not rely on task-specific external resources (unlike Noraset et al., 2017; Yang et al., 2019) or on pre-training (unlike Gadetsky et al., 2018).

Our contextual systems trained on the D_{Gad} dataset used the concatenation of the *definiendum* and the contextual cue as inputs. The definiendum was always at the start of the training example. This regular structure has shown to be useful for the models' performance: all models perform significantly worse on the more realistic data of D_{Ctx} than on D_{Gad}. The D_{Ctx} dataset is intrinsically harder for other reasons as well: it requires some form of lemmatization in every three out of eight training examples, and contains less data than other datasets, only half as many examples as D_{Nor}, and 20% less than D_{Gad}.

The surprisingly poor results of SELECT on the D_{Ctx} dataset may be partially blamed on the absence of a regular structure in D_{Ctx}. Unlike D_{Gad}, where the model must only learn to contextualize the first element of the sequence, in D_{Ctx} the model has to single out the *definiendum* which may appear anywhere in the sentence. Any information stored only in representations of contextual tokens will be lost to the decoders. The SELECT model therefore suffers of a bottleneck, which is highly regular in D_{Gad} and that it may therefore learn to cope with; however predicting *where* in the input sequence the bottleneck will appear is far from trivial in the D_{Ctx} dataset. We also attempted to retrain this model with various settings of hyperparameters, modifying dropout rate, number of warmup steps, and number of layers in the encoder—but to no avail. An alternative explanation may be that in the case of the D_{Gad} dataset, the regular structure of the input entails that the first positional encoding is used as an additive marking device: only *definienda* are marked with the positional encoding $\vec{\text{pos}}(1)$, and thus the architecture does not purely embrace a selective approach but a mixed one.

In any event, even on the D_{Gad} dataset where the margin is very small, the perplexity of the additive marking approach ADD is better than that of the SELECT model. This lends empirical support to our claim that definition modeling is a non-trivial sequence-to-sequence task, which can be

[6] Definition from Wiktionary.

better treated with sequence methods. The stability of the performance improvement over the non-contextual variant in both contextual datasets also highlights that our proposed additive marking is fairly robust, and functions equally well when confronted to somewhat artificial inputs, as in D_{Gad}, or to linguistically coherent sequences, as in D_{Ctx}.

6 Qualitative Analysis

filch	to seize
grammar	the science of language
implosion	a sudden and violent collapse

(a) Handpicked sample

sediment	to percolate
deputation	the act of inciting
ancestry	lineage

(b) Random sample

Table 2: Examples of production (non-contextual model trained on D_{Nor})

A manual analysis of definitions produced by our system reveals issues similar to those discussed by Noraset et al. (2017), namely self-reference,[7] POS-mismatches, over- and under-specificity, antonymy, and incoherence. Annotating distinct productions from the validation set, for the non-contextual model trained on D_{Nor}, we counted 9.9% of self-references, 11.6% POS-mismatches, and 1.3% of words defined as their antonyms. We counted POS-mismatches whenever the definition seemed to fit another part-of-speech than that of the *definiendum*, regardless of both of their meanings; cf. Table 2 for examples.

For comparison, we annotated the first 1000 productions of the validation set from our ADD model trained on D_{Ctx}. We counted 18.4% POS mismatches and 4.4% of self-referring definitions; examples are shown in Table 3. The higher rate of POS-mismatch may be due to the model's hardship in finding which word is to be defined since the model is not presented with the *definiendum* alone: access to the full context may confuse it. On the other hand, the lower number of self-referring definitions may also be linked to this richer, more varied input: this would allow the model not to fall back on simply reusing the *definiendum* as its own *definiens*. Self-referring definitions highlight that our models equate the meaning of the *definiendum* to the composed meaning of its *definientia*. Simply masking the corresponding output embedding might suffice to prevent this specific problem; preliminary experiments in that direction suggest that this may also help decrease perplexity further.

As for POS-mismatches, we do note that the work of Noraset et al. (2017) had a much lower rate of 4.29%: we suggest that this may be due to the fact that they employ a learned character-level convolutional network, which arguably would be able to capture orthography and rudiments of morphology. Adding such a sub-module to our proposed architecture might diminish the number of mistagged *definienda*. Another possibility would be to pre-train the model, as was done by Gadetsky et al. (2018): in our case in particular, the encoder could be trained for POS-tagging or lemmatization.

Lastly, one important kind of mistakes we observed is hallucinations. Consider for instance this production by the ADD model trained on D_{Ctx}, for the word "*beta*": "*the twentieth letter of the Greek alphabet (κ), transliterated as 'o'.*". Nearly everything it contains is factually wrong, though the general semantics are close enough to deceive an unaware reader.[8] We conjecture that filtering out hallucinatory productions will be a main challenge for future definition modeling architectures, for two main reasons: firstly, the tools and metrics necessary to assess and handle such hallucinations have yet to be developed; secondly, the input given to the system being word embeddings, research will be faced with the problem of grounding these distributional representations—how can we ensure that "*beta*" is correctly defined as "*the second letter of the Greek alphabet, transliterated as 'b'*", if we only have access to a representation derived from its contexts of usage? Integration of word embeddings with structured knowledge bases might be needed for accurate treatment of such cases.

[7] Self-referring definitions are those where a *definiendum* is used as a *definiens* for itself. Dictionaries are expected to be exempt of such definitions: as readers are assumed not to know the meaning of the *definiendum* when looking it up.

[8] On a related note, other examples were found to contain unwanted social biases; consider the production by the same model for the word "*blackface*": "*relating to or characteristic of the theatre*". Part of the social bias here may be blamed on the under-specific description that omits the offensive nature of the word; however contrast the definition of Merriam Webster for *blackface*, which includes a note on the offensiveness of the term, with that of Wiktionary, which does not. Cf. Bolukbasi et al. (2016); Swinger et al. (2018) for a discussion on biases within embedding themselves.

Error type	Context (*definiendum* in bold)	Production
POS-mismatch	her **major** is linguistics	most important or important
Self-reference	he wrote a letter of **apology** to the hostess	a formal expression of apology

Table 3: Examples of common errors (ADD model trained on D_{Nor})

7 Conclusion

We introduced an approach to generating word definitions that allows the model to access rich contextual information about the word token to be defined. Building on the distributional hypothesis, we naturally treat definition generation as a sequence-to-sequence task of mapping the word's context of usage (input sequence) into the context-appropriate definition (output sequence).

We showed that our approach is competitive against a more naive 'contextualize and select' pipeline. This was demonstrated by comparison both to the previous contextualized model by Gadetsky et al. (2018) and to the Transformer-based SELECT variation of our model, which differs from the proposed architecture only in the context encoding pipeline. While our results are encouraging, given the existing benchmarks we were limited to perplexity measurements in our quantitative evaluation. A more nuanced semantically driven methodology might be useful in the future to better assess the merits of our system in comparison to alternatives.

Our model opens several avenues of future explorations. One could straightforwardly extend it to generate definitions of multiword expressions or phrases, or to analyze vector compositionality models by generating paraphrases for vector representations produced by these algorithms. Another strength of our approach is that it can provide the basis for a standardized benchmark for contextualized and non-contextual embeddings alike: downstream evaluation tasks for embeddings systems in general either apply to non-contextual embeddings (Gladkova et al., 2016, eg.) or to contextual embeddings (Wang et al., 2019, eg.) exclusively, redefining definition modeling as a sequence-to-sequence task will allow in future works to compare models using contextual and non-contextual embeddings in a unified fashion. Lastly, we also intend to experiment on languages other than English, especially considering that the required resources for our model only amount to a set of pre-trained embeddings and a dataset of definitions, either of which are generally simple to obtain.

While there is a potential for local improvements, our approach has demonstrated its ability to account for contextualized word meaning in a principled way, while training contextualized token encoding and definition generation end-to-end. Our implementation is efficient and fast, building on free open source libraries for deep learning, and shows good empirical results. Our code, trained models, and data will be made available to the community.

Acknowledgments

We thank Quentin Gliosca for his many remarks throughout all stages of this project. We also thanks Kees van Deemter, as well as anonymous reviewers, for their thoughtful criticism of this work. The work was supported by a public grant overseen by the French National Research Agency (ANR) as part of the "Investissements d'Avenir" program: Idex *Lorraine Université d'Excellence* (reference: ANR-15-IDEX-0004).

References

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

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473.

Sergey Bartunov, Dmitry Kondrashkin, Anton Osokin, and Dmitry P. Vetrov. 2016. Breaking sticks and ambiguities with adaptive skip-gram. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics, AISTATS 2016, Cadiz, Spain, May 9-11, 2016*, pages 130–138.

Miroslav Batchkarov, Thomas Kober, Jeremy Reffin, Julie Weeds, and David Weir. 2016. A critique of word similarity as a method for evaluating distributional semantic models. In *Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP*, pages 7–12, Berlin, Germany. Association for Computational Linguistics.

Gemma Boleda. 2019. Distributional semantics and linguistic theory. *CoRR*, abs/1905.01896.

Tolga Bolukbasi, Kai-Wei Chang, James Y Zou, Venkatesh Saligrama, and Adam T Kalai. 2016. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 4349–4357. Curran Associates, Inc.

Tom Bosc and Pascal Vincent. 2018. Auto-encoding dictionary definitions into consistent word embeddings. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1522–1532. Association for Computational Linguistics.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2018. Universal sentence encoder. *CoRR*, abs/1803.11175.

Ting-Yun Chang, Ta-Chung Chi, Shang-Chi Tsai, and Yun-Nung Chen. 2018. xsense: Learning sense-separated sparse representations and textual definitions for explainable word sense networks. *CoRR*, abs/1809.03348.

Silvio Cordeiro, Carlos Ramisch, Marco Idiart, and Aline Villavicencio. 2016. Predicting the compositionality of nominal compounds: Giving word embeddings a hard time. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1986–1997, Berlin, Germany. ACL. CORE2018 rank: A*. `https://aclweb.org/anthology/P16-1187`.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G. Carbonell, Quoc V. Le, and Ruslan Salakhutdinov. 2019. Transformer-xl: Attentive language models beyond a fixed-length context. *CoRR*, abs/1901.02860.

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

J.R. Firth. 1957. *Papers in linguistics, 1934-1951*. Oxford University Press.

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

Anna Gladkova, Aleksandr Drozd, and Satoshi Matsuoka. 2016. Analogy-based detection of morphological and semantic relations with word embeddings: what works and what doesn't. In *SRWHLT-NAACL*.

Zellig Harris. 1954. Distributional structure. *Word*, 10(23):146–162.

Felix Hill, Kyunghyun Cho, and Anna Korhonen. 2016. Learning distributed representations of sentences from unlabelled data. *CoRR*, abs/1602.03483.

G. Klein, Y. Kim, Y. Deng, J. Senellart, and A. M. Rush. 2017. OpenNMT: Open-Source Toolkit for Neural Machine Translation. *ArXiv e-prints*.

Angeliki Lazaridou, Marco Arturo Marelli, and Marco Baroni. 2017. Multimodal word meaning induction from minimal exposure to natural text. *Cognitive science*, 41 Suppl 4:677–705.

Alessandro Lenci. 2018. Distributional models of word meaning. *Annual review of Linguistics*, 4:151–171.

Omer Levy and Yoav Goldberg. 2014a. Linguistic regularities in sparse and explicit word representations. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning*, pages 171–180. Association for Computational Linguistics.

Omer Levy and Yoav Goldberg. 2014b. Neural word embedding as implicit matrix factorization. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2177–2185. Curran Associates, Inc.

Edward Loper and Steven Bird. 2002. Nltk: The natural language toolkit. In *In Proceedings of the ACL Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics. Philadelphia: Association for Computational Linguistics*.

Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. 2017. Learned in translation: Contextualized word vectors. *CoRR*, abs/1708.00107.

Thanapon Noraset, Chen Liang, Lawrence Birnbaum, and Doug Downey. 2017. Definition modeling: Learning to define word embeddings in natural language. In *AAAI*.

Myle Ott, Sergey Edunov, David Grangier, and Michael Auli. 2018. Scaling neural machine translation. *CoRR*, abs/1806.00187.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proc. of NAACL*.

Alec Radford. 2018. Improving language understanding by generative pre-training.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Nathaniel Swinger, Maria De-Arteaga, Neil Thomas Heffernan IV, Mark D. M. Leiserson, and Adam Tauman Kalai. 2018. What are the biases in my word embedding? *CoRR*, abs/1812.08769.

Wilson Taylor. 1953. Cloze procedure: A new tool for measuring readability. *Journalism Quarterly*, 30:415–433.

Julien Tissier, Christophe Gravier, and Amaury Habrard. 2017. Dict2vec : Learning word embeddings using lexical dictionaries. In *EMNLP*.

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

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In the Proceedings of ICLR.

Tong Wang, Abdelrahman Mohamed, and Graeme Hirst. 2015. Learning lexical embeddings with syntactic and lexicographic knowledge. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 458–463. Association for Computational Linguistics.

Liner Yang, Cunliang Kong, Yun Chen, Yang Liu, Qinan Fan, and Erhong Yang. 2019. Incorporating sememes into chinese definition modeling. *CoRR*, abs/1905.06512.