VCDM: Leveraging Variational Bi-encoding and Deep Contextualized Word Representations for Improved Definition Modeling

Machel Reid **Edison Marrese-Taylor** **Yutaka Matsuo**
Graduate School of Engineering
The University of Tokyo
machelreid2004@gmail.com
{emarrese, matsuo}@weblab.t.u-tokyo.ac.jp

Abstract

In this paper we tackle the task of definition modeling, where the goal is to learn to generate definitions of words and phrases. Existing approaches for this task are discriminative, combining distributional and lexical semantics in an implicit rather than direct way. To tackle this issue we propose a generative model for the task, introducing a continuous latent variable to explicitly model the underlying relationship between a phrase used within a context and its definition. We rely on variational inference for estimation and leverage contextualized word embeddings for improved performance. Our approach is evaluated on four existing challenging benchmarks with the addition of two new datasets, CAMBRIDGE and the first non-English corpus ROBERT, which we release to complement our empirical study. Our Variational Contextual Definition Modeler (VCDM) achieves state-of-the-art performance in terms of automatic and human evaluation metrics, demonstrating the effectiveness of our approach.[1]

1 Introduction

In most current NLP tasks, fixed-length vector representations of words, word embeddings, are used to represent some form of the meaning of the word. In the case of humans, however, oftentimes we will use a sequence of words known as a *definition* —a statement of the meaning for a term— to express meanings of terms (words, phrases, or symbols). It is with this in mind that the question of "Can machines define?" is aimed to be answered with the task of **definition modeling** (Noraset et al., 2017).

Definition modeling can be framed as a task of conditional generation, in which the definition $\mathbf{d}$ of the word or phrase is generated given a conditioning variable $\mathbf{w}$ such as a word's associated word embedding or other representations of context. Current approaches for this task (Noraset et al., 2017; Gadetsky et al., 2018; Ni and Wang, 2017; Ishiwatari et al., 2019) are mainly encoder-decoder based, in which one encodes a contextual representation for a word/phrase $\mathbf{w}$ using a variety of features such as context or character composition, and uses the contextual representation(s) to generate the definition $\mathbf{d}$.

Despite the relative success of existing approaches for definition modelling, their discriminative nature —where distributional-derived information is at one end of the model and lexical information is at the other— limits their power as the underlying semantic representations of the distributional and lexical information are learned in an implicit rather than direct way. For example, although Ishiwatari et al. (2019) successfully showed that both local and global contexts are useful to disambiguate meanings of phrases in certain cases, their approach heavily relies on an attention mechanism to identify semantic alignments between the input phrase and the output definition, which may introduce noise and ultimately be insufficient to capture the entire meaning of each phrase-definition pair.

To tackle this issue, we propose to explicitly model the underlying semantics of phrase-definition pairs by introducing a continuous latent variable $\mathbf{z}$ over a definition space, which is used in conjunction with $\mathbf{w}$ to guide the generation of definition $\mathbf{d}$. The introduction of this latent representation enables us to treat it as a global defining signal during the generation process, complementing existing alignment mechanisms such as the attention.

Although the latent definition variable enables us to explicitly model underlying semantics of context-definition pairs, the incorporation of it into the task renders the posterior intractable. In this paper we recur to variational inference to estimate this in-

[1]We release the code at: https://github.com/machelreid/vcdm

Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, pages 6331–6344, November 16–20, 2020.

tractable posterior, effectively making our model a Conditional Variational Autoencoder and evolving the generation process from $p(\mathbf{d}|\mathbf{w})$ to $p(\mathbf{d}|\mathbf{w}, \mathbf{z})$.

We also note that existing approaches for definition modelling heavily rely on word embeddings, which due to their fixed nature can only capture so much of the semantics, being known to offer limited capabilities when dealing with polysemy. Considering the success of pretrained deep contextualized word representations which by specifically addressing these limitations have been shown to improve performance on a variety of downstream NLP tasks (Peters et al., 2018; Devlin et al., 2018), in this paper we propose a mechanism to integrate deep contextualized word representations in the definition modelling task. Specifically, we successfully leverage BERT (Devlin et al., 2018) as our contextual encoder and our definition encoder to produce representations for $\mathbf{w}$ and $\mathbf{d}$ respectively.

Finally, we develop two new datasets for this task, one derived from the Cambridge Dictionary [2], and the other derived from Le Petit Robert[3]. In summary, our contributions are:

- **Model:** We propose a novel approach for the task of definition modeling, leveraging deep contextualized word representation and the variational encoder-decoder architecture. We achieve new state-of-the-art performance on the definition modeling task, outperforming the previous state-of-the-art by as much as 9 BLEU points on the OXFORD dataset and 22 BLEU points on the ROBERT dataset.
- **Datasets:** We develop two new datasets CAMBRIDGE and ROBERT for this task. With ROBERT, a French dataset, being the first non-English dataset developed for this task.

Datasets and pre-trained models will be publicly released to the greater NLP community to help facilitate further advances on this task upon acceptance of this paper.

2 Related Work

Our work is related to the seminal paper by Hill et al. (2016), who proposed using the definitions found in everyday dictionaries as a means of bridging existing gaps between lexical and phrasal semantics. Effectively, they train a language model to map dictionary definitions to lexical representations of words, presenting the task of reverse dictionaries, where the goal is to return the name of a concept given a definition.

Noraset et al. (2017) later introduced the task of definition modeling, in which a model is tasked with generating a definition for a given word, given its respective embedding. The authors argued that, compared to other related tasks such as word similarity or analogical relatedness, definition generation can be considered a more transparent view of the information captured by an embedding. However, this method does not incorporate contextual information, preventing it from generating appropriate definitions for polysemic words. Addresing this, Gadetsky et al. (2018) studied the problem of polysemy in definition modeling, introducing an attention-based model which uses contextual information determine components in the embedding which may refer to a relevant word meaning.

Ni and Wang (2017) explore a different but related problem, proposing an approach for automatically explaining non-standard English expressions (i.e. slang) in a given sentence. They present a hybrid word-character sequence-to-sequence model that directly explains unseen non-standard expression, garnering reasonable definitions of expressions given their context.

More recently, Ishiwatari et al. (2019) have tackled some of the limitations of previous works on definition modelling and non-standard English expression explanation. Concretely, they note that whenever it is not possible to figure out the meaning of a given expression from its immediate local context, it is common to consult dictionaries for definitions or search documents or the web to find other global context to help in interpretation. In light of this, they introduce the task of describing a given phrase in natural language, based on its local and global contexts. To tackle this the authors introduce a model which consists of two context encoders (one for the local context, and one for the global context) as well as a description decoder. Our proposed model, uses a more practical variational encoder-decoder framework, allowing us to take advantage of explicitly modeling the phrase-definition relationship, while also leveraging deep contextualized word representations for more informative context representations.

[2] https://dictionary.cambridge.org/
[3] https://dictionnaire.lerobert.com/

Finally, our model is also related to Sohn et al. (2015), in which Conditional Variational Autoencoders (CVAEs) —an extension of the original variational autoencoder (VAE) (Kingma and Welling, 2014)— were proposed for generating diverse structured output, mainly in the context of image generation, and visual object segmentation and labeling. Our work is also related to CVAE models that have been developed for the domain of natural language processing, specifically Zhang et al. (2016) who proposed a CVAE in the context of Neural Machine Translation (NMT). As the usage of VAEs has become relatively common, we will omit a detailed explanation of these models, referring readers to Kingma and Welling (2014).

3 Proposed Approach

In a way analogous to previous work (Noraset et al., 2017), our proposed approach is a generative probabilistic model for word definitions, in which the goal is to estimate the probability of generating a definition d, given an input w. Concretely, we propose to directly capture the joint semantics of the (w, d) pairs by introducing a latent variable $\mathbf{z}$ to model the underlying *definition space*. Our proposed generative process can be formulated as follows:

$$p(\mathrm{d}|\mathrm{w}) = \int_z p(\mathrm{d}, z|\mathrm{w})d_z = \int_z p(\mathrm{d}|z, \mathrm{w})p(z|\mathrm{w})d_z \tag{1}$$

where the conditional probability $p(\mathrm{d}|\mathrm{w})$ evolves into $p(\mathrm{d}|\mathrm{w}, \mathrm{z})$, and the generation of the definition d is now conditioned on both the input variable w and our introduced continuous latent variable z.

Since the introduction of our latent variable makes posterior inference intractable, in this paper we resort to variational inference to perform posterior approximation. Effectively, this makes our proposed generative model a CVAE (Sohn et al., 2015) such that the variational lower bound can be formulated as follows:

$$\begin{aligned}\log p(\mathrm{d}|\mathrm{w}) \geq \mathbb{E}_{\mathrm{z}\sim q(\mathrm{z})}\big[\log p(\mathrm{d}|\mathrm{w}, \mathrm{z})\big] \\ - D_{\mathrm{KL}}\big[q(\mathrm{z})||p(\mathrm{z}|\mathrm{w})\big]\end{aligned} \tag{2}$$

where $q(\mathrm{z})$ is the introduced variational approximation to the intractable posterior $p(\mathrm{z}|\mathrm{w}, \mathrm{d})$ and $p(\mathrm{z}|\mathrm{w})$ is the prior distribution. Following previous work (Sohn et al., 2015; Zhang et al., 2016) we let $\mathbf{z}$, $\mathbf{w}$ and $\mathbf{d}$ be random vectors associated to z, w and d respectively, and utilize neural networks to estimate the following components.

- $q(\mathrm{z}) \approx q_\phi(\mathbf{z}|\mathbf{w}, \mathbf{d})$ is our variational approximation for the intractable posterior (the *recognition network*), which we model with a neural network with parameters ϕ. This makes $q_\phi(\mathbf{z}|\mathbf{w}, \mathbf{d})$ a *neural definition inferer*.
- $p(\mathrm{z}|\mathrm{w}) \approx p_\theta(\mathbf{z}|\mathbf{w})$ is a *(conditional) prior network*, parameterized by θ, which in our case can be regarded as a *neural definition prior*.
- $p(\mathrm{d}|\mathrm{w}, \mathrm{z}) \approx p_\theta(\mathbf{d}|\mathbf{w}, \mathbf{z})$ is a *generation network*, parameterized by θ which acts as a *variational definition modeler*.

In the following subsections we give details on how we specifically model each one of these components. With this in mind, we develop the following architecture comprised of 3 major components:

- **Encoders** - This component is comprised of two encoders - one *context encoder* to produce a representation for $\mathbf{w}$ and another *definition encoder* to produce a representation for $\mathbf{d}$ (Section 3.1).
- **Neural Definition Inferer** - This component *infers* the latent representation $\mathbf{z}$ from the representation of the word/phrase —the explicitly modeled prior $p_\theta(\mathbf{z}|\mathbf{w})$— and in conjunction with the definition (the approximated posterior $q_\phi(\mathbf{z}|\mathbf{d}, \mathbf{w})$) (Section 3.2).
- **Variational Definiton Modeler** - This component can be viewed as a decoder which takes in latent representation $\mathbf{z}$ to guide the generation of the target sentence, essentially $p_\theta(\mathbf{d}|\mathbf{w}, \mathbf{z})$ (Section 3.3).

Notation For clarity when explaining the approach, we define the notation conventions we will follow, namely: d refers to dimensions, $\boldsymbol{c}$ refers to the context vectors produced by the attention mechanism, g refers to projections/activations, and $\boldsymbol{h}$ refers to sets of vectors.

3.1 Encoders

3.1.1 Context Encoder

To encode the sequence in which the word in question is used, we adopt the BERT (Devlin et al., 2018) architecture, which is comprised of multiple Transformer (Vaswani et al., 2017) encoder layers pretrained on a masked-language modeling task to encode deep contextual word representations for a given sequence. BERT has also shown to be able to model the relationship between two tasks on pair-wise natural language understanding tasks. It is to this end that we propose the construction of *phrase-context pairs* to leverage this property in the context encoding process.

Inspired by *context-gloss pairs* (Huang et al., 2019) for the task of Word-Sense Disambiguation (WSD), we construct *phrase-context pairs* for our task of definition modeling. Often, there are differences in the word or phrase that we aim to define, and the lexeme form that is used in the context sentence. For example, the lemma *run* has the following forms: *run*, *runs*, *ran* and *running*, which all represent the same lexeme. To account for these discrepancies between the lemma and the lexeme form, we construct the aforementioned *phrase-context pairs*, which are constructed by simply inserting a separator token, denoted as [SEP], between the word/phrase and the context sentence. Below we show how this process would work for an example taken from Cambridge dictionary dataset for the lemma *leave*:

He *left* a wife and two children.
$\hookrightarrow$ *leave* [SEP] He *left* a wife and two children.

This form of construction for the *phrase-context pairs* comes with the added benefit of querying a sentence for a definition by simply prepending the word/phrase and a seperator token to the context sequence. As we use BERT as our encoder, we are able to leverage its self attentive nature to produce a representation of the word or phrase in question with respect to the context sentence.

As we initialize our context encoder with BERT, the phrase-context pair sequence $c = [w_t, \text{[SEP]}, c_2, \dots c_{M_c}]$, containing word or phrase w_t, is prepended by a [CLS] token and is appended by a [SEP] token, making $c_0 = \text{[CLS]}$ and $c_{M_c} = \text{[SEP]}$.

We define this **context encoder** as T_c, which takes in the context sequence c, and returns a sequence of annotation vectors for each token in c. We denote these annotation vectors as as $\boldsymbol{h}_c$, where $\{\boldsymbol{h}_c^{(i)}\}_{i=0}^{M_c} \in \mathbb{R}^{d_c}$ and

$$\boldsymbol{r}_{w_t} = T_c(c)[t] \tag{3}$$

is the t^{th} representation in h_c, representing w_t. In the case that w_t is split into multiple subword tokens by the BERT tokenizer, we set the word representation to be the mean of each of its subword representations. Namely, in the case that w_t is comprised of the n^{th} to the m^{th} subtokens,

$$\boldsymbol{r}_{w_t} = \frac{1}{m-n} \sum_{i=n}^{m} T_c(c)[i] \tag{4}$$

3.1.2 Definition Encoder

The definition encoder, which we denote as T_d, is also initialized with BERT. This encoder takes in the definition sequence $d = [d_0, d_1, \dots, d_{M_e}]$ as input and represents d as:

$$\boldsymbol{r}_d = T_d(d)[0] \tag{5}$$

where $\boldsymbol{r}_d \in \mathbb{R}^{d_e}$. We take the representation (corresponding to the preprended [CLS] token) as a representation for the entire definition sequence.

3.2 Neural Definition Inferer

We formulate the posterior distribution $q_\phi(\mathbf{z}|\mathbf{d}, \mathbf{w})$ and prior distribution $p_\theta(\mathbf{z}|\mathbf{w})$ as multivariate Gaussians with a diagonal covariance matrices. To model these distributions we make use of neural networks, following Zhang et al. (2016).

3.2.1 Neural Definition Posterior

As modeling the true posterior $p(\mathbf{z}|\mathbf{d}, \mathbf{w})$ is generally intractable, to approximate this true posterior, we use a variational distribution, formulated as the following multivariate Gaussian:

$$q_\phi(\mathbf{z}|\mathbf{d}, \mathbf{w}) = \mathcal{N}(\mathbf{z}; \mu(\mathbf{d}, \mathbf{w}), \sigma(\mathbf{d}, \mathbf{w})^2 \mathbf{I}) \tag{6}$$

which is parameterized by the mean and $\mu(\mathbf{d}, \mathbf{w})$ standard deviation $\sigma(\mathbf{d}, \mathbf{w})$, both which are treated as functions of definition $\mathbf{d}$ and phrase $\mathbf{w}$ parameterized by neural networks.

From the neural encoding mechanisms, we gather the definition representation $\boldsymbol{r}_d$, and the context representation $\boldsymbol{r}_{w_t}$. We then concatenate $\boldsymbol{r}_d$ and $\boldsymbol{r}_{w_t}$ and project the resulting vector onto our latent space, setting $\boldsymbol{h}_z = g(\boldsymbol{W}_z[\boldsymbol{r}_{w_t}; \boldsymbol{r}_d] + \boldsymbol{b}_z)$, where $\boldsymbol{W}_z \in \mathbb{R}^{d_z \times (d_e + d_c)}$ is a trainable weight matrix, $\boldsymbol{b}_z \in \mathbb{R}^{d_z}$ is a trainable bias vector and $g(\cdot)$ represents a non-linearity activation. In our experiments we set $g(\cdot)$ to be the *tanh*$(\cdot)$ activation function, following previous work.

To attain the aforementioned mean and variance vectors parameterizing the variational distribution, setting $\boldsymbol{\mu} = \boldsymbol{W}_\mu \boldsymbol{h}_z + \boldsymbol{b}_\mu$ and $\log \boldsymbol{\sigma}^2 = \boldsymbol{W}_\sigma \boldsymbol{h}_z + \boldsymbol{b}_\sigma$, where $\boldsymbol{W}_\mu, \boldsymbol{W}_\sigma \in \mathbb{R}^{d_z \times d_z}$ are trainable weight matrices parameterizing the projection and $\boldsymbol{b}_\mu, \boldsymbol{b}_\sigma \in \mathbb{R}^{d_z}$ are bias vectors.

In order to make the parameters θ differentiable for gradient descent optimization, we use the "reparameterization trick" (Kingma and Welling, 2014) setting $\boldsymbol{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \cdot \boldsymbol{e}$, where $\boldsymbol{e} \sim \mathcal{N}(0, \boldsymbol{I})$ is a noise variable sampled from a multivariate Gaussian distribution to derive our latent vector $\boldsymbol{z}$.

3.2.2 Neural Definition Prior

Our prior is a conditional distribution formulated as the following multivariate Gaussian:

$$p_\theta(\mathbf{z}|\mathbf{w}) = \mathcal{N}(\mathbf{z}; \mu'(\mathbf{w}), \sigma'(\mathbf{w})^2 \mathbf{I}) \tag{7}$$

which is parameterized by $\mu'(\cdot)$, and $\sigma'(\cdot)$ which are both solely functions of phrase $\mathbf{w}$. In a similar fashion to the *neural definition posterior*, we make use of a linear projection to project $\boldsymbol{r}_{w_t}$ to the mean vector $\boldsymbol{\mu}'$ and another linear projection to derive the log variance vector. During inference (at test time) when sampling from $p_\theta(\mathbf{z}|\mathbf{w})$, we set our latent vector $\boldsymbol{z}$ to be the mean vector $\boldsymbol{\mu}'$.

To initialize the decoding procedure detailed in the next subsection, we feed the latent representation $\boldsymbol{z}$ and project it to the decoding space setting $\boldsymbol{h}'_d = g(\boldsymbol{W}_d \boldsymbol{z} + \boldsymbol{b}_d)$, where $\boldsymbol{W}_d \in \mathbb{R}^{d_d \times d_z}$ and $\boldsymbol{b}_d \in \mathbb{R}^{d_d}$.

3.3 Variational Definition Modeler

Given phrase $\mathbf{w}$ and latent representation $\mathbf{z}$, the process of definition modeling can be formulated as the following conditional language model:

$$p(\mathbf{d}|\mathbf{w}, \mathbf{z}) = \textstyle\prod_{j=1}^{M_d} p(d_j | d_{<j}, \mathbf{z}, \mathbf{w}) \tag{8}$$

$$p(d_j | d_{<j}, \mathbf{z}, \mathbf{w}) = g_d(\boldsymbol{s}_j, \boldsymbol{c}_j) \tag{9}$$

where g_d is a feed-forward neural network which returns a distribution over the elements in the decoder vocabulary given the context vector $\boldsymbol{c}_j$ (see Eq. 16) and decoder state $\boldsymbol{s}_j$.

During generation of the definition sequence, we want the decoder to rely on all of the encoded components at each timestep. We modify the LSTM Cell (Hochreiter and Schmidhuber, 1997) to encompass previous context vector $\boldsymbol{c}_{j-1}$, and the projected latent definition representation $\boldsymbol{h}'_d$.

Intuitively, at each timestep j, we want the generated token to have the ability to rely on each of these components in the case that the previous hidden state and/or generated token does not provide enough information or misleads the accurate generation of the next token. We refer to this modified cell as the Variational Contextual Definition Modeler (VCDM) Cell, and the resulting decoder as a VCDM-RNN. The VCDM Cell calculates the decoder hidden state $\boldsymbol{s}_j$ as follows[4]:

$$\boldsymbol{i}_j = \sigma(\boldsymbol{W} E_{d_j} + \boldsymbol{U} \boldsymbol{s}_{j-1} + \boldsymbol{A} c_{j-1} + \boldsymbol{V} \boldsymbol{h}'_d) \tag{10}$$

$$\boldsymbol{f}_j = \sigma(\boldsymbol{W}_f E_{d_j} + \boldsymbol{U}_f \boldsymbol{s}_{j-1} + \boldsymbol{A}_f \boldsymbol{c}_{j-1} + \boldsymbol{V}_f \boldsymbol{h}'_d) \tag{11}$$

$$\boldsymbol{o}_j = \sigma(\boldsymbol{W}_o E_{d_j} + \boldsymbol{U}_o \boldsymbol{s}_{j-1} + \boldsymbol{A}_o \boldsymbol{c}_{j-1} + \boldsymbol{V}_o \boldsymbol{h}'_d) \tag{12}$$

$$\tilde{\boldsymbol{C}}_j = g(\boldsymbol{W}_g E_{d_j} + \boldsymbol{U}_g \boldsymbol{s}_{j-1} + \boldsymbol{A}_g \boldsymbol{c}_{j-1} + \boldsymbol{V}_g \boldsymbol{h}'_d) \tag{13}$$

$$\boldsymbol{C}_j = \sigma(\boldsymbol{f}_j \cdot \boldsymbol{C}_{j-1} + \boldsymbol{i}_j \cdot \tilde{\boldsymbol{C}}_j) \tag{14}$$

$$\boldsymbol{s}_j = g(\boldsymbol{C}_j) \cdot \boldsymbol{o}_j \tag{15}$$

where $E_{d_j} \in \mathbb{R}^{d_w}$ is the embedding for the target word, $\boldsymbol{W}, \boldsymbol{W}_f, \boldsymbol{W}_o, \boldsymbol{W}_g \in \mathbb{R}^{d_d \times d_w}$, $\boldsymbol{U}, \boldsymbol{U}_f, \boldsymbol{U}_o, \boldsymbol{U}_g \in \mathbb{R}^{d_d \times d_d}$, $\boldsymbol{A}, \boldsymbol{A}_f, \boldsymbol{A}_o, \boldsymbol{A}_g \in \mathbb{R}^{d_d \times d_d}$, and $\boldsymbol{V}, \boldsymbol{V}_f, \boldsymbol{V}_o, \boldsymbol{V}_g \in \mathbb{R}^{d_d \times d_d}$ are trainable weight matrices parameterizing the RNN cell.

Additionally, at each decoder timestep j we attend to the set of annotation vectors $\boldsymbol{h}_c$ produced by the last layer of the context encoder. To compute context vector $\boldsymbol{c}_j$, we use general attention (Luong et al., 2015) shown below:

$$\boldsymbol{c}_j = \textstyle\sum_{i=1}^{T} \alpha_i \boldsymbol{h}_c^{(i)} \tag{16}$$

$$\alpha_i = \text{softmax}(\boldsymbol{s}_j^\top \boldsymbol{W}_a \boldsymbol{h}_c^{(i)}) \tag{17}$$

where $\boldsymbol{W}_a \in \mathbb{R}^{d_d \times d_c}$, and α_i can be viewed as an alignment over $\boldsymbol{h}_c$ and $\boldsymbol{c}_j$ as a vector capturing the encoder hidden states scaled by this alignment.

3.4 Optimization challenges

Despite the VAE's appeal as a tool to learn unsupervised representations through the use of latent variables, these models are often found to ignore latent variables when using powerful generators. To overcome this issue of "posterior collapse" (Bowman et al., 2016), we incorporate the following heuristics: (1) annealing the KL term from 0 to 1 using a sigmoid annealing schedule, following Bowman et al. (2016) and (2) thresholding the KL term in the objective function with a constant λ using the "free bits" technique (Kingma et al., 2016). With these changes, our objective function is modified

[4]Note: For clarity, we omit the bias terms in Equations 10-15

CAMBRIDGE	Count		Length		
Partition	**Phrases**	**Examples**	**Phrase**	**Context**	**Definition**
Train	21,993	42,689	1.01	9.14 ± 4.27	11.64 ± 6.75
Valid	4,671	5,335	1.00	9.25 ± 4.24	11.69 ± 6.80
Test	4,670	5,337	1.00	9.20 ± 4.32	11.60 ± 6.75
Overall	24,557	53,361	1.01	9.16 ± 4.27	11.64 ± 6.76
ROBERT	**Count**		**Length**		
Partition	**Phrases**	**Examples**	**Phrase**	**Context**	**Definition**
Train	30,049	71,073	1.00	10.51 ± 7.36	7.97 ± 4.95
Valid	6,992	8,884	1.00	10.55 ± 7.47	7.93 ± 4.90
Test	6,985	8,884	1.00	10.46 ± 7.36	8.03 ± 5.00
Overall	33,507	88,842	1.00	10.51 ± 7.38	7.97 ± 4.94

Table 1: Statistics for CAMBRIDGE and ROBERT. The number of individual phrases, number of examples, and the mean and s.d. of the lengths of each partition of the dataset are reported

to become the following:

$$\mathcal{L}(\theta, \phi) = -\mathbb{E}_{\mathbf{z} \sim q_\phi(\mathbf{z}|\mathbf{w},\mathbf{d})}\left[\log p_\theta(\mathbf{d}|\mathbf{w},\mathbf{z})\right] + \gamma \sum_i \max(\lambda, D_{\mathrm{KL}}(q_\phi(\mathbf{z}_i|\mathbf{w},\mathbf{d})||p(\mathbf{z}_i|\mathbf{w}))) \quad (18)$$

Where γ denotes the annealing term that follows the sigmoid schedule and λ denotes the *target rate*, and the sub-index i denotes the i^{th} dimension of the latent vector $\mathbf{z}$. In our experiments, we set the total $\lambda = 1$.

4 Empirical Study

4.1 Data

To evaluate our approach we make use of the following previously released datasets: OXFORD (Gadetsky et al., 2018) built from Oxford Dictionaries[5], URBAN built from the Urban Dictionary[6], and WIKIPEDIA (Ishiwatari et al., 2019) built from Wikipedia.

The task of definition modeling with respect to each of the aforementioned datasets can be regarded as three separate domains, in which (1) OXFORD can be viewed as a corpus of "traditional" dictionary definitions, where most common words in a given language are contained, (2) URBAN can be viewed as a corpus of "uncommon , slang words in which one often has to use context and subword information to decipher the meaning, and (3) WIKIPEDIA can be viewed almost as a description generation task of named entities, conditioned on the given context.

[5] oxforddictionaries.com
[6] urbandictionary.com

In addition to these datasets, we also develop the CAMBRIDGE (English) and ROBERT (French) dataset. We collect this data from the online version of the Cambridge Dictionary[7] and Le Petit Robert[8]. Following the spirit of previously released datasets, we include three components for each example: (1) the word or phrase being defined, (2) an example (context sentence) in which it is contained and (3) its corresponding definition. These datasets can be seen as an addition to the domain of "traditional" dictionary definitions, with ROBERT being the first non-English dataset. Please refer to Table 1 for statistics regarding these datasets.

4.2 Experiments

4.2.1 Our Model: VCDM

We initialize each of our encoders with BERT-base-uncased (or in the case of ROBERT, CamemBERT-base (Martin et al., 2019)), setting $d_e, d_c = 768$. We set latent dimension $d_z = 83$, and the LSTM decoder's hidden size $d_d = 512$ with an output vocabulary size of 10k, initializing embeddings with Word2Vec (Mikolov et al., 2013). We perform gradient descent using the Adam optimizer (Kingma and Ba, 2014) with its default hyperparameters. During decoding, we use the beam-search algorithm, setting the beam size to 5. We implement all models in PyTorch (Paszke et al., 2019).

4.2.2 Baselines

Local and Global Context-Aware Description generator (LoG-CAD): proposed by Ishiwatari et al. (2019), this model achieved the previous state-of-the-art on existing datasets for this task. The model makes use of a BiLSTM (Graves and Schmidhuber, 2005) to encode sentence-level context, a character-level CNN (Zhang et al., 2015) to encode character-level information, and pretrained Google CBOW[9](Mikolov et al., 2013) vectors (for ROBERT we use the French `fasttext` word vectors (Grave et al., 2018)). During decoding, this method makes use of a 2-layer attentional 300-dim LSTM decoder with an additional gating mechanism to combine all these sources of encoding information.

LSTM baseline (LSTM): To show the effect of continous latent variable modeling for this task,

[7] dictionary.cambridge.org
[8] https://dictionnaire.lerobert.com/
[9] https://code.google.com/archive/p/word2vec/

and for a more direct comparison to LoG-CAD, we implement an LSTM version of our proposed architecture. Following LoG-CAD, use a 2-layer 300 dimensional BiLSTM as each encoder and use a 10k Byte-Pair tokenized (Sennrich et al., 2016) encoder vocabulary. The *neural definition inferer* and the *variational definition modeler* are kept the same as our proposed method.

BERT Baselines: This baseline is a single-layer attentional 512-dim LSTM-LM decoder conditioned on $\boldsymbol{r}_{w_t}$. We use two variants: (1) **BERT-fr** where $\boldsymbol{r}_{w_t}$ is produced by a a frozen BERT-base encoder and (2) **BERT-ft** where $\boldsymbol{r}_{w_t}$ is produced by a BERT-base encoder finetuned during training.

4.3 Evaluation

When comparing our approach to our baselines we make use of two automatic evaluation metrics, namely sentence-level BLEU (Papineni et al., 2002; Koehn et al., 2007) and the recently proposed BERTScore (Zhang et al., 2019). While the former is a well-known metric for machine translation, based mainly on n-gram matching between source and target, the latter is a rather new approach that leverages BERT's pretrained contextual embeddings, matching words in candidate and reference sentences by way of cosine similarity. Concretely, BERTScore computes 3 metrics, namely precision (denoted as P_{BERT}), recall (denoted as R_{BERT}) and F1 score (denoted as F_{BERT}).

Our interest in BERTScore sparks from the fact that it has been recently shown to correlate better with human judgement in system evaluations, and to address the potential issue of coherent definition generations being given low evaluation scores as a result of having zero or low n-gram overlap with the reference sentence.

Finally, in addition to our automatic evaluation we also performed a human study, where three different human annotators evaluated the output generated by our proposed approach, as well as by the LoG-CAD and BERT-ft baselines. We followed the approach by Ishiwatari et al. (2019) and used their 1-5 scale:

1. Completely wrong or self-definition
2. Correct topic with wrong information
3. Correct but incomplete
4. Small details missing
5. Correct

to evaluate 100 randomly sampled instances from OXFORD.

DATA	Model	*BLEU*	P_{BERT}	R_{BERT}	F_{BERT}
OXFORD	LoG-CAD	18.63	86.40	80.57	83.38
	LSTM	21.02	85.58	85.51	85.52
	BERT-fr	18.26	85.95	85.11	85.50
	BERT-ft	27.26	87.36	87.07	87.19
	VCDM	**27.38**	**87.47**	**87.11**	**87.27**
URBAN	LoG-CAD	10.65	78.73	81.77	80.09
	LSTM	11.10	84.27	83.54	83.87
	BERT-fr	9.89	84.04	82.36	83.12
	BERT-ft	11.45	84.91	82.65	83.71
	VCDM	**13.90**	**85.15**	**83.70**	**84.36**
WIKIPEDIA	LoG-CAD	36.65	89.51	88.17	88.83
	LSTM	38.86	90.09	88.44	89.21
	BERT-fr	35.97	89.51	88.11	88.77
	BERT-ft	**42.97**	90.48	**89.54**	**89.97**
	VCDM	42.27	**90.89**	88.97	89.87
CAMBRIDGE	LoG-CAD	16.87	86.09	85.32	85.68
	LSTM	16.44	86.21	85.43	85.81
	BERT-fr	17.90	87.17	85.95	86.53
	BERT-ft	20.04	87.81	86.88	87.24
	VCDM	**22.46**	**88.16**	**87.46**	**87.70**
ROBERT	LoG-CAD	22.94	69.77	68.09	68.80
	LSTM	39.76	78.89	79.18	78.90
	BERT-fr	23.61	73.74	71.90	72.63
	BERT-ft	41.50	81.82	80.54	81.02
	VCDM	**44.97**	**82.80**	**81.96**	**82.24**

Table 2: Results on the test set for URBAN, OXFORD, WIKIPEDIA, CAMBRIDGE, and ROBERT.

To compare the values obtained for each example across two models, we utilized t-tests and pair-wise bootstrap resampling tests with 10,000 samples (Koehn, 2004), controlling for the random seed (set to 2 in our experiments).

5 Results

Automatic Evaluation: Table 2 shows the results on the test set for each reported metric and dataset. Firstly, we note that the LSTM Baseline is able to consistently outperform LoG-CAD in terms of BERTScore, although with mixed results in terms of BLEU. We think this difference is mainly due to the n-gram matching nature of BLEU, which tends to give better scores for longer but incorrect generations, as the example in Table 3 shows, while also being unable to adequately handle cases where the definitions are expressed using words not present in the gold standard. We believe these results validate the usage of a metric such as BERTScore on this task, ultimately showing that tackling definition

Word	Frankenstein
Context	In arming the dictator, the US was creating a ***Frankenstein***
Reference	something that destroys or harms the person or people who created it

Generated	BL	P	R	F
something that you say or do that you think someone of something is ridiculous	**12.5**	83.41	84.78	84.09
an extremely frightening or offensive person	8.13	**87.00**	84.78	**85.88**

Table 3: An example from the CAMBRIDGE test set, showing an evaluation issue caused by BLEU. The generated outputs of the LoG-CAD baseline are shown above, and ours below. BL stands for sentence BLEU and P, R and F stand for P_{BERT}, R_{BERT}, and F_{BERT}.

Configuration	$\mathbf{F_{BERT}}$ **(Δ)**	**BLEU (Δ)**
VCDM	**87.70** (—)	**22.46** (—)
Decoder LSTM Cell	87.68 (-0.02)	22.27 (-0.19)
Frozen definition encoder	87.14 (-0.56)	20.70 (-1.76)
Tied encoders	87.14 (-0.56)	20.59 (-1.87)
Frozen encoders	86.35 (-1.35)	17.42 (-5.04)
Frozen context encoder	85.19 (-2.51)	13.49 (-8.97)

Table 4: Results of the ablation study performed on CAMBRIDGE.

modeling with a generative approach can lead to improved results, and suggesting that the incorporation of a latent variable that models the underlying definition space is beneficial for this task.

Results on Table 2 also show that the inclusion of BERT significantly improves generation quality in terms of BERTScore and BLEU on most datasets. This suggests that the inclusion of pretrained deep contextual word representations is beneficial for the task, which is expected given its contextual nature. We also see that VCDM is able to successfully leverage BERT, as our model is able to offer improved results compared to BERT baselines in all datasets except WIKIPEDIA. We think these results offer additional empirical evidence to support the effectiveness of our generative approach. Improvements provided by our model are particularly significant in the case of URBAN, a dataset which there are many rare words and the context is arguably less informative due to its noisy properties.

We surmise that the subpar performance of VCDM over BERT-ft in WIKIPEDIA is related to the properties of the dataset domain (i.e. description generation of named entities). With this in mind, it could be argued that a completely context- focused

Model	Model	p-value bootstrap	p-value t-test
VCDM	LoG-CAD	0.005	1.4×10^{-8}
BERT-ft	LoG-CAD	0.006	4.2×10^{-5}
VCDM	BERT-ft	0.797	1.6×10^{-2}

Table 5: Exact p-values of the performed statistical tests, to compare the scores obtained during our human evaluation.

architecture (such as that of our finetuned BERT baseline) has properties that are more beneficial in this setting. Contrary to findings in Ishiwatari et al. (2019) which argue for the inclusion of a global context during generation, we find that a contextually-focused (local context) architecture with a strong context encoder (such as BERT) results in better performance within this domain.

Ablation Study: To further evaluate the contribution of each introduced component in our approach we performed an ablation study on CAMBRIDGE. Results of these experiments are summarized in Table 4, where it is possible to see that each of our introduced components is beneficial to the task. Note that the VCDM-Cell vs LSTM-Cell improvement is minimal on this dataset. The purpose of the integration of the latent variable in the decoder LSTM cell is for it to act as a ”global definition signal” so we can rely on the properties of the latent variable. As this property is especially useful in cases in which there is noisy context, we think it is reasonable to assume that as context here is more informative, the performance gain from including a global definition signal is relatively small. We also see that freezing the context encoder has a extremley negative impact on performance. We believe this is because the context-encoder hasn't effectively learned to use the *phrase-context pairs* (Sec. 3.1.1).

Human Evaluation: Average human scores obtained are 2.51 for LoG-CAD, 3.08 for BERT-ft and **3.31 for VCDM**. When tested for statistical significance (Table 5), we observed that both VCDM and BERT-ft were superior to LoG-CAD with 99% confidence, using both paired t-tests or pair-wise bootstrap resampling tests (Koehn, 2004), and that the difference between VCDM and BERT-ft was statistically significant at 95% for the t-test.

Qualitative Evaluation: Finally we provide a qualitative evaluation by showing an example of the output of our model and of two of our base-

Word:	Present(VB)	Present(NN)
Context:	Within a sexist ideology and a male-dominated cinema, the woman is ***presented*** as what she represents for man.	In addition to this, think of the ***presents***, the toys, gift sets, and most importantly, all that wrapping paper.
Reference:	To represent (someone or something) to others in a particular way	A thing given to someone as a gift
LoG-CAD:	a person who is present in a particular way	a person's mind
BERT-ft:	Portray or regard (someone) as a particular person, idea or action	An item of furniture presented to resemble a bride <unk>
VCDM:	To portray or describe (someone or something) in a particular context	A thing kept as a gift for children

Table 6: Example showing the generated definitions for two senses of the word "present", taken from OXFORD.

lines, in Table 6. In the OXFORD dataset, which this example is taken from there are 7 senses of the "present", showing VCDM's ability to effectively disambiguate between a large amount of senses.

6 Conclusion

In this paper we have introduced a generative model that directly combines distributional and lexical semantics via a continuous latent variable for the task of definition modeling. Empirical results on multiple corpora, including two new datasets released, show that our model is able to outperform previous work by a consistent margin, also successfully being able to leveraging contextualized word representations. For future work we are interested in exploring how definition modeling could be adapted to a multilingual or cross-lingual setting.

Acknowledgments

We are grateful for the support provided by the NVIDIA Corporation, donating two of the GPUs used for this research. We thank Victor Zhong for insightful discussions, and thank Pablo Loyola, Cristian Rodriguez-Opazo, and Jorge Balazs for proofreading the work and providing useful suggestions.

References

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio. 2016. Generating Sentences from a Continuous Space. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 10–21, Berlin, Germany. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of deep bidirectional transformers for language understanding.

Artyom Gadetsky, Ilya Yakubovskiy, and Dmitry P. Vetrov. 2018. Conditional generators of words definitions. *CoRR*, abs/1806.10090.

Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomas Mikolov. 2018. Learning word vectors for 157 languages. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*.

Alex Graves and Jürgen Schmidhuber. 2005. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural networks*, 18(5-6):602–610.

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

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Luyao Huang, Chi Sun, Xipeng Qiu, and Xuanjing Huang. 2019. Glossbert: Bert for word sense disambiguation with gloss knowledge. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.

Shonosuke Ishiwatari, Hiroaki Hayashi, Naoki Yoshinaga, Graham Neubig, Shoetsu Sato, Masashi Toyoda, and Masaru Kitsuregawa. 2019. Learning to Describe Unknown Phrases with Local and Global Contexts. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3467–3476, Minneapolis, Minnesota. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization.

Diederik P. Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. *arXiv:1312.6114 [cs, stat]*. ArXiv: 1312.6114.

Durk P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. 2016. Improved Variational Inference with Inverse Autoregressive Flow. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 4743–4751. Curran Associates, Inc.

Philipp Koehn. 2004. Statistical Significance Tests for Machine Translation Evaluation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 388–395, Barcelona, Spain. Association for Computational Linguistics.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *ACL*.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*.

Louis Martin, Benjamin Muller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric Villemonte de la Clergerie, Djamé Seddah, and Benoît Sagot. 2019. Camembert: a tasty french language model.

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

Ke Ni and William Yang Wang. 2017. Learning to Explain Non-Standard English Words and Phrases. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 413–417, Taipei, Taiwan. Asian Federation of Natural Language Processing.

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

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8026–8037. Curran Associates, Inc.

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

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Kihyuk Sohn, Honglak Lee, and Xinchen Yan. 2015. Learning Structured Output Representation using Deep Conditional Generative Models. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 3483–3491. Curran Associates, Inc.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need.

Biao Zhang, Deyi Xiong, Jinsong Su, Hong Duan, and Min Zhang. 2016. Variational Neural Machine Translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 521–530, Austin, Texas. Association for Computational Linguistics.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with BERT. *CoRR*, abs/1904.09675.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in neural information processing systems*, pages 649–657.

VCDM: Leveraging Variational Bi-encoding and Deep Contextualized Word Representations for Improved Definition Modeling

A Datasets

Tables 1, 2 and 3 provide a summary of the sizes of each partition for the datasets OXFORD, URBAN and WIKIPEDIA, respectively.

Partition	Count		Length		
	Phrases	Examples	Phrase	Context	Definition
Train	33,128	97,855	1.00	17.74	11.02
Valid	8,867	12,232	1.00	17.80	10.99
Test	8,850	12,232	1.00	17.56	10.95

Table 1: Statistics for OXFORD. The number of individual phrases, number of examples, and the mean lengths of each partition of the dataset are reported.

Partition	Count		Length		
	Phrases	Examples	Phrase	Context	Definition
Train	190,696	411,384	1.54	10.89	10.99
Valid	26,876	57,883	1.54	10.86	10.95
Test	26,875	38,371	1.68	11.14	11.50

Table 2: Statistics for URBAN. The number of individual phrases, number of examples, and the mean lengths of each partition of the dataset are reported.

Partition	Count		Length		
	Phrases	Examples	Phrase	Context	Definition
Train	151,995	887,455	2.10	18.79	5.89
Valid	8,361	44,003	2.11	19.21	6.31
Test	8,397	57,232	2.10	19.02	6.94

Table 3: Statistics for WIKIPEDIA. The number of individual phrases, number of examples, and the mean lengths of each partition of the dataset are reported.

B Evaluation

We make use of the `sentence-bleu.cpp`[1] script in the MOSES (Koehn et al., 2007) GitHub repository to compute sentence-level BLEU, and use the `bert-score` Python package[2] to calculate BERTScore, with hash `roberta-large_L17_no-idf_version=0.3.2 (hug_trans=2.8.0)` for the English datasets and hash `bert-base-multilingual-cased_L9_no-idf_version=0.3.3(hug_trans=2.10.0)` for ROBERT which is in French. For datasets where there are multiple examples for a given word sense, such as WIKIPEDIA, we note that results provided by (Ishiwatari et al., 2019) are obtained by first averaging the evaluation metrics for multiple examples for a given sense —although this is not reported on the paper— which tends to inflate the final values of the metrics. Instead, in this paper report example-wise metric averages, which provide more realistic values of the aggregated evaluation metrics.

C Model Details

We train each of our models using a batch size of 64, and set 1e-3 as the initial learning rate for the Adam optimizer. However, when finetuning BERT (or CamemBERT) in any circumstance, we set the initial learning rate for the BERT parameters to be 5e-5 and use a linear warmup schedule, warming up for the first epoch. We train all models in PyTorch (Paszke et al., 2019), and use the HuggingFace[3] (Wolf et al., 2019) implementation of CamemBERT-base and BERT-base-uncased.

Additionally, we re-implement LoG-CAD (Ishiwatari et al., 2019) using the authors' GitHub repository[4].

[1] `https://github.com/moses-smt/mosesdecoder/blob/master/mert/sentence-bleu.cpp`

[2] `https://github.com/Tiiiger/bert_score`

[3] `https://github.com/huggingface/transformers`

[4] `https://github.com/shonosuke/ishiwatari-naacl2019`

D Infrastructure and Environment

Experiments for different datasets were run in two different machines:

- A server machine with an Intel Xeon E5-2630 CPU, and two NVIDIA RTX-2080 (Driver 418.56, CUDA 10.1) GPUs, running Ubuntu 16.04
- An additional server machine with an Intel Core i7-6850K CPU and two NVIDIA Titan Xp (Driver 430.50, CUDA 10.1) GPUs, also running Ubuntu 16.04

E Additonal Output Examples

Word:	Yen
Context:	If Koizumi has enjoyed some economic success, say critics, it has been through a combination of good luck and what many believe has been an artificial weakens of they yen against the dollar.
Reference:	the basic monetary unit of japan.
LoG-CAD:	a longing or yearning
BERT-ft:	a monetary in a foreign country
VCDM:	the basic monetary unit of japan, equal to 100 cents.

Table 4: Descriptions for the rare word “yen"

Word:	doucheturd
Context:	Brad, you’re such a doucheturd.
Reference:	insulting noun, being both a douche or douchebag and a turd
LoG-CAD:	a person who is a douchebag
BERT-ft:	a person who is a douchebag
VCDM:	a person who is a mix of a douche and a turd

Table 5: Descriptions for the slang word “doucheturd"

Gold	(especially of a political party) sponsor (a candidate) in an election
LoG-CAD	a candidate or candidate in a race or election
VCDM	set (a person or team) in an election
Gold	(of a batsman) run from one wicket to the other in scoring or attempting to score a run .
LoG-CAD	a race or contest in which a race is run
VCDM	(of a sports team) run by hitting at the reach of the three runs
Gold	a large open stretch of land used for pasture or the raising of stock
LoG-CAD	(of a horse) be <unk>
VCDM	a specially <unk> area for cattle or cattle
Gold	a large open stretch of land used for pasture or the raising of stock
LoG-CAD	(of a horse) be <unk>
VCDM	a specially <unk> area for cattle or cattle
Gold	a preliminary test of a procedure or system
LoG-CAD	a person or thing that is <unk> or <unk>
VCDM	a continuous search or undertaking
Gold	a race between candidates for elective office
LoG-CAD	a series of people who are <unk>
VCDM	be the charge of
Gold	a row of unravelled stitches
LoG-CAD	a short , narrow <unk>
VCDM	a <unk> in a careless or <unk> way
Gold	a track made or regularly used by a particular animal
LoG-CAD	move or cause to move in a specified direction
VCDM	a <unk> or <unk> run over a tree
Gold	become undone
LoG-CAD	be <unk>
VCDM	become undone by being undone
Gold	cause something to pass or lead somewhere
LoG-CAD	move or move in a <unk>
VCDM	cause something to pass or lead somewhere by constant strength
Gold	cause to perform
LoG-CAD	make a series of facts or plans
VCDM	put in an area
Gold	diarrhoea .
LoG-CAD	a person 's <unk>
VCDM	a <unk> of <unk>
Gold	emit or exude a liquid
LoG-CAD	(of a person 's eyes) move or cause to move in a specified direction
VCDM	(of a person ' s <unk>) become <unk> with <unk>
Gold	extend or continue for a certain period of time
LoG-CAD	be a result of
VCDM	pass for a certain time of time
Gold	fail to stop at (a red traffic light)
LoG-CAD	(of a vehicle) move or move in a specified direction
VCDM	run at or at a particular <unk>
Gold	move about in a hurried and hectic way
LoG-CAD	be in a specified way
VCDM	go or run somewhere in a particular place
Gold	move or cause to move between the spools of a recording machine
LoG-CAD	move or move in a specified direction
VCDM	use or enable (a container) to a desired point in a specified direction
Gold	publish or be published in a newspaper or magazine
LoG-CAD	a <unk> or <unk>
VCDM	(of a newspaper or a public station) publish or broadcast (a television programme)
Gold	put (a form of public transport) in service
LoG-CAD	(of a vehicle) be <unk> or <unk>
VCDM	provide (an undertaking , train , or service) for a service
Gold	the act of running ; traveling on foot at a fast pace
LoG-CAD	a run in a race
VCDM	the act of running ; traveling on foot at a fast pace
Gold	the act of testing something
LoG-CAD	the act of <unk> something
VCDM	the act of testing something
Gold	travel a route regularly
LoG-CAD	move or move in a specified direction
VCDM	travel a route regularly
Gold	the after part of a ship 's bottom where it rises and narrows towards the stern .
LoG-CAD	a <unk> or <unk> .
VCDM	an act of <unk> a boat ' s foot
Gold	the average or usual type of person or thing
LoG-CAD	a person 's existence or belief
VCDM	the general aspects of something , especially a language

Table 6: Comparison of the outputs for the 23 senses of “run’ on OXFORD.

References

Shonosuke Ishiwatari, Hiroaki Hayashi, Naoki Yoshinaga, Graham Neubig, Shoetsu Sato, Masashi Toyoda, and Masaru Kitsuregawa. 2019. Learning to Describe Unknown Phrases with Local and Global Contexts. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3467–3476, Minneapolis, Minnesota. Association for Computational Linguistics.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *ACL*.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8026–8037. Curran Associates, Inc.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing.