Acta Linguistica Academica 69 (2022) 4, 450–473
DOI: 10.1556/2062.2022.00584

BiVaSE: A bilingual variational sentence encoder with randomly initialized Transformer layers

BENCE NYÉKI*

Hungarian Research Centre for Linguistics, Hungary

Received: May 3, 2022 • Accepted: October 2, 2022

ABSTRACT

Transformer-based NLP models have achieved state-of-the-art results in many NLP tasks including text classification and text generation. However, the layers of these models do not output any explicit representations for texts units larger than tokens (e.g. sentences), although such representations are required to perform text classification. Sentence encodings are usually obtained by applying a pooling technique during fine-tuning on a specific task. In this paper, a new sentence encoder is introduced. Relying on an autoencoder architecture, it was trained to learn sentence representations from the very beginning of its training. The model was trained on bilingual data with variational Bayesian inference. Sentence representations were evaluated in downstream and linguistic probing tasks. Although the newly introduced encoder generally performs worse than well-known Transformer-based encoders, the experiments show that it was able to learn to incorporate linguistic information in the sentence representations.

KEYWORDS

sentence embedding, variational autoencoder, Transformer, pre-training, pooling

1. INTRODUCTION

Modeling natural language semantics is arguably a complex task that is hard to solve with computational methods. In fact, language modeling in natural language processing (NLP) is based on solving a simpler task. More specifically, a language model (LM) is essentially a function p that assigns probabilities to sequences of symbols. If $s = (s_1, \ldots, s_n)$ is such a sequence

* Corresponding author. E-mail: nyeki.bence@nytud.hu

consisting of n symbols, then its probability is usually calculated as a product of conditional probabilities:

$$p(s) = \prod_{i=1}^{n} p(s_i | s_1, \dots, s_{i-1}) \tag{1}$$

Although technologies and model architectures have been rapidly changing over the last decades, this definition of an LM has remained unaltered, see e.g. Bengio et al. (2003) and Radford et al. (2019).

Some successful models, such as Word2Vec by Mikolov, Chen et al. (2013) and Mikolov, Sutskever et al. (2013), were designed to predict probability distributions for masked symbols given their context. This task, which is slightly different from language modeling, is usually referred to as masked language modeling (MLM). However, models trained on MLM still exploit the same basic idea: consider some symbols from a context, then predict another symbol. One obvious application of predicting probability distributions is automatic text generation by sampling from these distributions. For instance, the GPT-3 model of OpenAI by Brown et al. (2020) achieved state-of-the-art results in many NLP tasks related to text generation.

This, however, does not mean that NLP models have nothing to do with modeling natural language semantics. The research in this field usually relies on the assumption that if a model θ (which is actually a set of parameters) is trained to solve the language modeling task (or a similar task), then it will be possible to use a subset of θ to map sequences of symbols to vector representations that can be interpreted as "meaning". This assumption was motivated by the work of Harris (1954) and Firth (1957) frequently referred to as distributional semantics, according to which the meaning of a word can be described in terms of co-occurrences with other words.

This research aspect is important from the viewpoint of linguistics. Mel'chuk (1974), for example, defined his main research objective as creating $\ll$ Meaning $\Leftrightarrow$ Text $\gg$ models, i.e. finding mappings between surface text representations (sequences of graphical or audible units) and semantic structures.

Furthermore, it should be noted that if language as a system is to be modelled with computational and statistical methods, one will face a problem that is hard to overcome: while a large amount of data that encodes surface text representations is available, meaning is generally not observable.[1] The research published by Mikolov, Chen et al. (2013) and Mikolov, Sutskever et al. (2013) was one of the most important works that showed that reasonable vector representations of word meanings, called word embeddings, can be trained efficiently using neural networks. The question how these computer-generated word embeddings can represent certain aspects of semantics can also be studied using more recent and sophisticated models, see e.g. Mickus et al. (2019).

However, it is obvious that the research should not be restricted to word embeddings. Finding representations of larger text units can be useful not only at a theoretic level, as many real world applications require compact sentence or text representations, such as sentiment analysis, semantic similarity search, topic modeling, etc. Sentence embeddings (i.e. vectorized sentence meaning representations) are usually obtained by applying mathematical operations to word vectors, such as averaging. In this paper, several neural network models capable of

[1]The word "generally" is used because one might argue that neurolinguistic methods actually provide us with some tools to study natural language meaning directly in the human brain, but the data collected with such methods is not easily accessible in a large amount and hard to work with within a non-neurolinguistic framework.

outputting fixed-size sentence embeddings[2] are described and evaluated. A new bilingual variational sentence encoder is introduced. This model relies on variational auto-encoding, as well as the simple average pooling technique to construct sentence embeddings. The main focus of evaluation is on choosing some aspects of natural language semantics and empirically showing to what extent the models can capture these aspects. Results on sentiment analysis tasks, which are more closely related to real-world applications, are also reported. For the experiments, English and Hungarian language data were used.

The main contribution of this work includes:

- Pre-training a model with the variational auto-encoding objective with randomly initialized weights.
- Comparing the performances of neural network models on different tasks (linguistic probing and sentiment analysis).
- Releasing source code.[3]

2. RELATED WORK

A brief overview of related work is provided in this section.

2.1. Sentence embedding models trained on monolingual data

It is natural to start the review of related work with English models. There are many datasets for model training and testing in English and significant novelties in model architecture or training methods are always evaluated on such datasets. As a result, the evolution of English models also reflects the overall development of NLP.

As stated in Section 1, averaging word embeddings is a simple way of obtaining sentence embeddings, but more complex techniques have also been elaborated.

Recurrent neural networks (RNNs) relying on the long short-term memory (LSTM) architecture introduced by Hochreiter & Schmidhuber (1997) or gated recurrent units (GRUs) proposed by Chung et al. (2014) allow to straightforwardly extract sentence embeddings from a sequence-processing network. RNNs gradually update their hidden state h over T time steps: T is the total number of atomic elements (tokens[4]) in the input sequence and the vector representations (w^1, …, w^T) of these elements are fed to the RNN one time step each. The value of the hidden state vector h^t outputted at time step t depends on h^{t-1} and w^t. The last hidden state vector h^T can be expected to encode the meaning of the whole sequence.

The Skip-Thought models introduced by Kiros et al. (2015) made use of RNNs. Their architecture included an encoder RNN that generated a latent representation for an input sentence

[2]Here, the term "sentence" is not used with scientific rigour. The text passages that will be referred to as sentences in this paper are actually multiple sentences or fragments of what linguists may call sentences. This indefiniteness can be explained with the large amount of data required for training neural networks: large data collections are inherently noisy and therefore a dataset of sentences should be regarded rather as a dataset of objects "close to sentences".

[3]https://github.com/nytud/ae-sentence-embeddings

[4]Tokens can be words as well as smaller meaningless superficial text segments, even graphemes.

s_i ($i \in \{2, 3, \ldots\}$) and two decoder RNNs: the first was trained to predict the sentence s_{i-1} preceding s_i and the second was trained to predict the next sentence s_{i+1}. In order to minimize the prediction error, the encoder had to compress semantic and syntactic information in the latent sentence representations. This training objective requires continuous text pieces of at least three sentences, but no human-annotated data is needed.

Conneau et al. (2017) proposed the InferSent method, i.e. training sentence embedding models on the Stanford Natural Language Inference (SNLI) dataset introduced by Bowman et al. (2015). This is an example of supervised learning as the SNLI dataset consists of labelled sentence pairs. The labels indicate the relation between the two sentences: entailment (B follows from A), contradiction (B contradicts to A) or neutral (neither entailment nor contradiction). The authors found that the encoder models trained on SNLI generalize well to other tasks. Their best performing model was a bidirectional LSTM (BiLSTM), which is an RNN that processes sequences not only from the beginning to the end but also from the opposite direction. The sentence embeddings outputted by this model were 4,096 dimensional vectors. This is a relatively large embedding size compared to the dimensionality of embeddings returned by other models. For example, Word2Vec word vectors usually consist of a few hundreds of elements.

The power means embedding method by Rücklé et al. (2018) was a reaction to InferSent. A sentence s was encoded by concatenating different power mean values of the vector representations of the words included in s.

Vaswani et al. (2017) released the Transformer, which achieved new state-of-the-art results in machine translation. The Transformer architecture is based on an encoder and a decoder, none of which uses recurrent units. It was showed that the so-called attention mechanism can effectively capture semantic relatedness between words in text sequences without step-by-step processing. Further research pointed out that the encoder and decoder modules can also be separately trained to handle a variety of tasks: for example, refer to Devlin et al. (2018) and Radford & Narasimhan (2018) for more details on the Transformer-based BERT encoder and the GPT decoder, respectively. A disadvantage of the Transformer models is that they do not produce any latent representations that can be easily interpreted as sentence or text embeddings. The output of a Transformer encoder is a matrix of contextualized token representations.[5]

Cer et al. (2018) introduced two versions of the Universal Sentence Encoder (USE), one of which was based on the Transformer architecture. The USE applies the following method to obtain sentence embeddings: it calculates the element-wise sum of the token vectors returned by the Transformer encoder and then normalizes the sum by dividing it by the square root of the number of tokens in the input sequence. The pre-training of the USE included both unsupervised (Skip-Thought) and supervised (natural language inference) tasks. The size of its sentence embeddings is 512.

Reimers & Gurevych (2019) released their Sentence-Transformer models that heavily rely on transfer learning. This means that Transformer encoder models with already pre-trained weights were fine-tuned (trained further) on new supervised tasks. Rather than taking the normalized sum of contextualized token embeddings, the authors applied max, CLS and average pooling. More details on pooling techniques are provided in Section 3.2. The training approach adopted

[5]Contextualized means that the same token can be mapped to different representations in different contexts.

by the Sentence-Transformers led to high-quality sentence embeddings that can effectively capture semantic similarity.

In general, model architectures initially tested on English data can also be applied to a large number of other languages. From the viewpoint of the present paper, Hungarian models are also of interest apart from English ones.

For Hungarian, Nemeskey (2020) and Nemeskey (2021) introduced huBERT, which is a robust pre-trained encoder that achieved state-of-the-art results in named entity recognition and potentially can be used as a sentence encoder.

2.2. Multilingual and cross-lingual sentence encoders

The goal of developing multilingual and cross-lingual sentence encoders is to support the automatic processing of a variety of natural languages. As many languages of the world lack large datasets, the concepts of zero-shot and few-shot learning become especially important. Zero-shot learning refers to training a model on a general task and then evaluating it on new tasks without leveraging additional knowledge from task-specific examples. Few-shot learning is a similar technique, but it allows the models to rely on a small amount of examples in order to learn to solve a new problem.

The LASER model by Artetxe & Schwenk (2019) supports 93 languages. Its language-agnostic sentence embeddings were trained on translation tasks and evaluated on cross-lingual sentence classification. The experiments conducted with LASER illustrate zero-shot learning. One of these experiments included training a classifier head on top of the pre-trained sentence embeddings on the English data of the XNLI (cross-lingual natural language inference) dataset created by Conneau, Rinott et al. (2018). Sentences in other languages from the dataset were classified with the same sentence encoder and classifier head without further training.

The results of a study of few-shot learning were described by Bari, Haider & Mansour (2021), who applied a simple nearest neighbors method to sentence embeddings outputted by a multilingual encoder in order to perform cross-lingual classification.

Few-shot and zero-shot text classification is not the only domain where multilingual models are applicable. State-of-the-art results on bitext mining (e.g. collecting parallel sentence pairs from monolingual corpora) were reported by Feng et al. (2020) who introduced LaBSE (language-agnostic BERT sentence embeddings). A Transformer encoder was pre-trained with MLM and translation language modeling (TLM) proposed by Conneau & Lample (2019). The latter method is similar to MLM, but the target probability distribution of a masked token *t* is conditioned on not only a monolingual context of *t* but also on the translation of the sequence in which *t* occurs. High-quality sentence embeddings were achieved by fine-tuning the pre-trained model on translation ranking, which implies choosing the translation of a sentence in language **A** from a set of sentences in language **B**. The cosine similarity score between sentence embeddings was used to rank translations.

It should also be mentioned that one of the best pre-trained multilingual models is XLM-R by Conneau et al. (2019). This is a Transformer encoder trained with MLM on 100 languages. XLM-R demonstrated especially effective cross-lingual transfer with a small amount of labelled validation data for each language on which the model was evaluated. As XLM-R is a Transformer encoder, it does not output explicit sentence embeddings, but pooling techniques can be applied to its token embeddings to obtain vectors that represent larger text units.

2.3. Variational encoders

The models discussed so far did not make use of variational methods. VAE models, briefly described in Section 3.1, are relatively rarely applied in NLP, although they can provide a regularized sentence embedding space. The most important effect of training sentence encoders with the VAE objective is that the sentence representations outputted by the encoder are not discrete points in an n-dimensional embedding space but the parameters of n-dimensional (usually Gaussian) probability distributions. One may call such a probability distribution a *cloud* (e.g. Gaussian cloud) in an n-dimensional space where the center of the cloud is the region where most of the probability density is concentrated. Most tasks, however, require sentence representations to be vectors. This can be solved by sampling from the probability distributions. The regularization comes from the constraint that the probability distributions must be as close to a prior distribution as possible.

Bowman et al. (2016) and Li et al. (2019) trained RNN-based variational models on English data. Zhang et al. (2016) also used RNNs to create a variational neural machine translation system. In more recent research done by Li et al. (2020), a Transformer-based encoder and decoder were integrated into the Optimus network pre-trained with the VAE objective on a large amount of English data.

The research in this fields usually concentrates on evaluating the overall VAE perplexity and the latent space. The focus of this work is on experiments aimed at finding out if the VAE objective affects the quality of the sentence embeddings pooled from a pre-trained Transformer encoder.

2.4. Datasets and evaluation

Sentence embeddings are usually evaluated on various classification tasks which can be downstream tasks related to real-world applications, problems concerning natural language understanding such as NLI or semantic textual similarity (STS) and linguistic probing tasks. The latter are aimed at verifying whether sentence embeddings can capture specific semantic or syntactic information such as verb tense or semantic integrity. 11 linguistic probing tasks are described in detail by Conneau, Kruszewski et al. (2018). Perone et al. (2018) evaluated several sentence encoders on tasks from each of the above mentioned categories and found that the models tended to specialize in certain groups of tasks and could not achieve the same results over the whole collection of evaluation datasets.

The study of Eger, Rücklé & Gurevych (2019) can be used as a practical guideline when evaluating sentence representations. An important observation is that comparing sentence embeddings of different sizes can be misleading. As larger vectors can contain more information, a model **A** with embedding size n can achieve significantly better test results than a model **B** with embedding size $m < n$ even without training objective or architecture differences.

Some of the most important English datasets supporting sentence encoder evaluation are SentEval by Conneau & Kiela (2018) and the datasets included in the GLUE benchmark Wang et al. (2018).

There are also some Hungarian datasets with sentence-level annotation. Ligeti-Nagy et al. (2022) recently released the HuLu benchmark, which is aimed at creating a resource for Hungarian that is comparable with GLUE. Apart from HuSST,[6] the sentiment subcorpus of HuLu, other sentiment corpora also exist, such as OpinHuBank created by Miháltz (2013) and the Hungarian Twitter Sentiment Corpus.[7]

3. BACKGROUND

This section provides a minimal theoretical background necessary to understand the methods outlined in Section 4.

3.1. Variational autoencoders

Autoencoders are latent variable models composed of an encoder and a decoder. When an input data point x is fed to the model, the encoder first maps x to a latent representation z and then the decoder tries the reconstruct x given z. The goal of such a seemingly meaningless training objective is to find a reasonable mapping between the input space X and the latent space Z. This simple architecture scheme is depicted in Figure 1.

If a latent variable model is to be trained on natural language data, any input x is a surface text representation and z can be considered a corresponding meaning representation that, in the ideal case, captures as much syntactic and semantic information as possible. However, the latent space of a simple autoencoder is usually highly irregular and uninterpretable. Minimizing the reconstruction loss guarantees only that the decoder finds a mapping between Z and X that reasonably fits the training data.

Variational autoencoders (VAEs) proposed by Kingma & Welling (2014) and Rezende, Mohamed & Wierstra (2014) regularize Z using Bayesian methods. In this theoretical framework, Z is considered a random variable about which one may make prior assumptions. Putting a standard Gaussian prior $\mathcal{N}(0; 1)$ on Z is a common choice: this reflects the prior assumption that Z usually takes values close to the origin of the latent space. It is briefly explained below how this prior is used to regularize the latent space during training.

A Bayesian approach involving a latent variable requires the training process to maximize the likelihood of the data X conditioned on the latent variable and the model parameters. The optimal model parameters that would maximize the likelihood cannot usually be calculated analytically, but even in these cases it is possible to find a lower bound of the likelihood. It can be proved that the so-called *evidence lower bound* (ELBO) for a VAE can be written as given in (2), provided that the data consists of N data points $(x^{(1)}, \ldots, x^{(N)})$.

$$\mathscr{L}(\theta, \phi; \mathbf{X}) = \sum\nolimits_{i=1}^{N} -D_{KL}(q_\phi(\mathbf{z}|\mathbf{x}^{(i)})\|p_\theta(\mathbf{z})) + \mathbb{E}_{q_\phi(z|\mathbf{x}^{(i)})}\left[\log p_\theta(\mathbf{x}^{(i)}|\mathbf{z})\right] \tag{2}$$

In (2), ϕ and θ denote the encoder and decoder parameters respectively, q_ϕ and p_θ are probability distributions. The left term on the right hand side of the equation is the negative

[6]https://huggingface.co/datasets/NYTK/HuSST

[7]http://opendata.hu/dataset/hungarian-twitter-sentiment-corpus

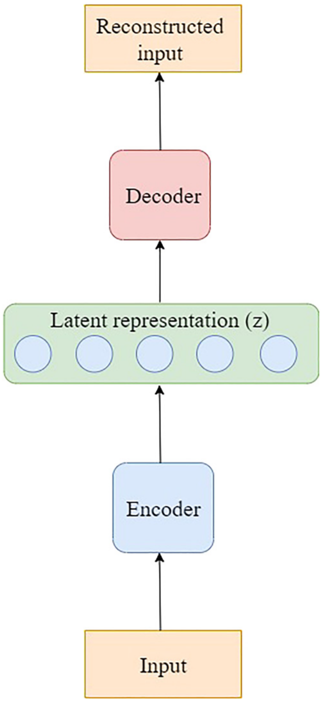

Figure 1. The autoencoder architecture scheme

Kullback-Leibler (KL) divergence between the probability distribution of the latent variable conditioned on the data and the prior put on Z. This is normally interpreted as a regularization loss. The conditional distribution is outputted by the encoder. The right term is a negative reconstruction loss.

This means that the ELBO is maximized when both the reconstruction loss and the KL divergence are minimized. If the KL divergence is calculated analytically, it is possible to simply add it to the reconstruction loss and use gradient descent to update ϕ and θ. Consequently, the model will be encouraged to encode the inputs as probability distributions close to the prior distribution of the latent variable in terms of KL divergence. Then a sample from the predicted distribution is passed to the decoder whose task is to reconstruct the input. In other words, the training goal is to deviate from the prior as little as possible and, at the same time, reconstruct the input as well as possible.

The choice of the prior before training is important. Theoretically, it should be a distribution that resembles the true but unknown posterior distribution $p_\theta(\mathbf{z}|\mathbf{x})$, i.e. the probability distribution of the latent variable given the data and the decoder parameters, but a standard Gaussian is a reasonable default choice in most cases.

This algorithm is expected to lead to a regularized latent space, i.e. most inputs will be represented as Gaussian "clouds" around its origin. Some straightforward advantages are:

- **Easy data generation.** In order to generate new data with high likelihood, it is sufficient to sample from a standard Gaussian distribution and pass the sampled value to the decoder.
- **Easy outlier detection.** If the encoder returns Gaussian parameters distant from the parameters of the standard normal distribution, it is likely that the input data point is noise.

A VAE can be regarded as a regularized autoencoder where regularization is provided by the KL divergence. Li et al. (2020) state that that this interpretation naturally suggests that the KL divergence term can be scaled by a constant multiplier β:

$$\mathscr{L}_{\beta} = \beta\mathscr{L}_{R} + \mathscr{L}_{E} \tag{3}$$

In (3), $\beta\mathscr{L}_R$ is the scaled regularization loss with $\beta > 0$, $\mathscr{L}_E$ is the reconstruction loss and $\mathscr{L}_\beta$ is the overall loss.

A well-known problem that frequently arises during VAE training is the so-called *posterior collapse*. It refers to a training state when the model converges to a local optimum by letting the encoder rapidly approach the uninformative prior distribution and treating its outputs as random noise ignored by the decoder. Several heuristics have been proposed to overcome this problem, such as annealing the KL loss multiplier during training as applied by Bowman et al. (2016) or putting a threshold on the KL loss as proposed by Kingma et al. (2016). The threshold is applied to each component of the latent vector representation provided by the encoder in order to make the model give up trying to reduce the regularization loss once the pre-defined minimum (i.e. the threshold) is reached. Li et al. (2019) found that pre-training the VAE without the regularization loss and then applying the KL threshold method leads to efficient training on text data. The intuition behind the first phase is that the decoder is less likely to ignore the encoder outputs if it provides informative representations from the beginning of the second phase when the regularization loss appears.

It is a natural intuition that a regularized embedding space may be interpreted from a linguistic viewpoint: “typical” or “unmarked” sentence meanings are centered around a semantic origin, while syntactically or semantically ill-formed sentences are far from the origin.

3.2. Pooling

In the context of Transformer-based sentence encoders, pooling is an operation that allows to obtain a vector representation of a sentence from a matrix of contextualized token embeddings without adding parameters to the model. To describe these operations, the following notation will be used: $M \in \mathbb{R}^{SxH}$ is a matrix with S rows and H columns; each row is a contextualized token embedding, thus H is the embedding size. Reimers & Gurevych (2019) experimented with three pooling techniques:

- **MAX.** In each column of M, the maximal value is taken.
- **Average or mean.** The arithmetic mean of the rows of M is taken.
- **CLS.** BERT uses the metatoken CLS prepended to beginning of every input sequence. So the first row m_1 of M is the embedding of a token that is present in every input. As the embeddings are contextualized, it is expected that m_1 can reflect the meaning of the whole sequence.

Reimers & Gurevych (2019) use the mean pooling technique as default, although they do not confirm that it is superior to the alternatives. In the original BERT paper by Devlin et al. (2018), the CLS pooling technique was recommended. LaBSE was also trained with CLS pooling. In practice, both of these two techniques tend to work well.

It is worth noticing that all of the above mentioned pooling operations keep the embedding size unchanged: if the size of the token embeddings is H, the size of the sentence vectors is also H.

4. METHODS AND MODELS

In this section, the theoretical considerations concerning the creation of a new sentence encoder are discussed. The other models it was compared with are also mentioned in Subsection 4.2.

4.1. Bilingual variational sentence encoder

In this subsection, a new bilingual variational sentence encoder (BiVaSE) and its training objective are introduced. The creation of the encoder was motivated by the following hypotheses:

- **Latent space regularization supports classification.** A common type of linguistic probing and downstream tasks is sequence classification: the encoder outputs a sequence representation (in our case, it is a sentence embedding) and then a label from a finite set is assigned to it. The classification output is usually provided by a feed-forward neural network layer that was not pre-trained with the encoder and thus its weights need to be tuned to solve a concrete task. When a classification model is being trained, the encoder weights are usually fine-tuned to achieve a better fit of the concrete dataset.[8] Training only the classifier head weights is generally not sufficient. However, if the latent embedding space is regularized, training only the classifier head might result in better metric scores. Even if full fine-tuning is necessary to achieve satisfying results, the regularization may still make the convergence faster.
- **A Transformer encoder can be trained from scratch using the VAE objective.** A Transformer model is typically pre-trained with the MLM or LM objective. In this work, it is attempted to train a Transformer encoder as part of a VAE autoencoder, i.e. as an encoder that learns a probability distribution $q_\phi(z|\mathbf{x}^{(i)})$ for any $\mathbf{x}^{(i)}$ data point from the training dataset. Table 1 underlines the differences between the pre-training strategies.

Table 1. Differnences between pre-training strategies. Pre-training in NLP is considered the training phase when a model learns general language features. It usually implies training a randomly initialized model on a single task, but it is split into two subphases in the case of Optimus

Model/architecture	LM/MLM pre-training	VAE pre-training
BERT	✓	✗
Optimus	✓	✓
BiVaSE	✗	✓

[8]This is the concept of transfer learning.

Relying on these hypotheses, the BiVaSE architecture and training objective were defined as described below.

A BERT encoder was initialized with a vocabulary size of 42,000; all other hyperparameters followed the standard BERT base configuration.[9] The average pooling technique was used to pool a sequence representation from the contextualized embeddings.

In each forward pass after the pooling operation, the mean and standard deviation parameters of a Gaussian distribution were obtained from the sentence embedding. The KL divergence between the resulting distribution $q_\phi(x)$ and the prior standard Gaussian $p_\theta(z)$ were calculated as a loss value, then a vector v was sampled from $q_\phi(x)$. This vector v was passed to the decoder, which is a 2-layer deep GRU, in order to calculate the reconstruction loss. The sampled v vector was passed to the first layer of the GRU as its initial state. The initial state of the second layer was calculated as the average of the last hidden state of the first layer and v. The embedding layer was shared between the encoder and the decoder. The embeddings and the RNN layers were connected with a feed-forward layer with *tanh* activation. The scheme of the model architecture is depicted in Figure 2.

This architecture defines an asymmetric autoencoder: the encoder contains more weights than the decoder.

Optimus introduced by Li et al. (2020) is also a Transformer-based VAE, but there are two major differences between it and BiVaSE: *(i)* Optimus consists of a BERT encoder and a GPT-2 decoder. *(ii)* Before training with the VAE objective, the weights of Optimus were initialized with the weights of BERT and GPT-2. The weights of BiVaSE were randomly initialized. As a consequence of *(i)*, passing a sample v from a distribution predicted by the encoder to the decoder is less straightforward in Optimus than in BiVaSE. Li et al. (2020) resolved this issue with latent vector injection, which can mean either adding a special memory vector to the GPT-2 layer inputs or updating the decoder token embeddings with information from v.

Making the model multilingual was motivated by the question whether a Transformer-based VAE can converge and its encoder can be fine-tuned if its pre-training task is to process inputs in multiple languages.

One more question that needed to be addressed before training BiVaSE concerns tokenization, i.e. the process of segmenting input texts to atomic units. As all weights of BiVaSE were initialized randomly, it did not depend on any pre-trained model and a new tokenizer could be created for it. A BPE tokenizer with a vocabulary size of 42,000 was trained on a sample from the pre-training dataset (described in Section 5). Byte pair encoding was originally proposed as an information compression technique by Gage (1994). In the recent years, the algorithm has been adapted to text data and widely applied to train subword tokenizers.

4.2. Competitors

BiVaSE is compared with an English monolingual, a Hungarian monolingual and a multilingual model. These are the cased BERT model,[10] huBERT[11] and XLM-R,[12] respectively.

[9]Please refer to Devlin et al. (2018) for details.

[10]https://huggingface.co/bert-base-cased

[11]https://huggingface.co/SZTAKI-HLT/hubert-base-cc

[12]https://huggingface.co/xlm-roberta-base

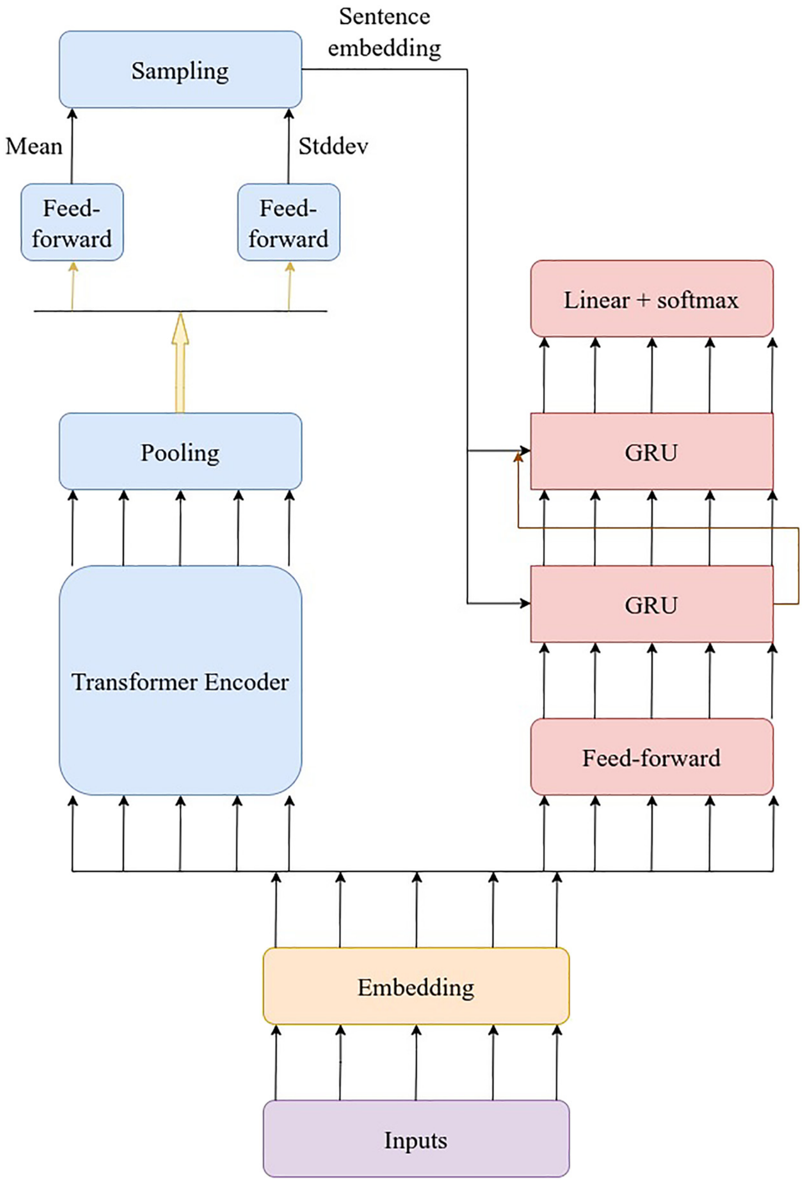

Figure 2. The BiVaSE architecture. The blue color indicates the encoder layers and the red color indicates the decoder layers

The monolingual models are evaluated only on datasets of their "native" languages, while XLM-R and BiVaSE are evaluated on all datasets.

By default, the layers of pre-trained models are frozen during evaluation and only their classifier heads are allowed to adapt to the specific tasks. This method provides us with a better understanding of what aspects of sentence meaning the models learned to capture during pre-training than full fine-tuning.[13] However, results with full fine-tuning are also reported for completeness in Section 7.

As BiVaSE is a VAE, it is necessary to explain how it was connected with a classifier head. The decoder part was dropped, and the Gaussian mean vectors outputted by the model were used as sentence representations, which were passed to the classifier head. The KL loss was ignored during full fine-tuning, allowing the latent space to arbitrarily change in order to contribute to a better fit on the data. When only the classifier heads were trained, the KL loss did not have any effect as the Transformer encoder layers were frozen.

In this work, each classifier head consisted of a single neuron. Applying the sigmoid activation to the classifier output is equivalent to logistic regression.

5. THE DATA

5.1. Pre-training data

BiVaSE was not trained on parallel sentences but the training corpus was created from English-Hungarian parallel corpora.[14] The data sources were Hunglish v2.0 by Varga et al. (2005), ParaCrawl 9[15] and further English-Hungarian corpora available through the OPUS service.[16] Opus was introduced by Tiedemann (2012) to provide open access to many parallel datasets. The detailed list of the corpora downloaded from Opus is provided in the Appendix.

After downloading and merging the corpora, the resulting text collection was preprocessed with the Bifixer and Bicleaner tools introduced by Sánchez-Cartagena et al. (2018) and Ramírez-Sánchez et al. (2020). The merged corpus was deduplicated and normalized with Bifixer, then its output was filtered with Bicleaner: sentence pairs scoring lower than 0.4 were dropped. The remaining sentence pairs were split into single sentences and the data was shuffled. This resulted in 172,877,288 sentences which were tokenized before training BiVaSE. Sequences longer than 128 tokens were truncated.

5.2. Fine-tuning data

The models were fine-tuned and evaluated on four types of binary text classification tasks. These are linguistic acceptability, sentiment analysis and two linguistic probing tasks described by Conneau, Kruszewski et al. (2018): SOMO (semantic odd man out) and Verb Tense.

[13]More precisely, this can elucidate which syntactic or semantic features can be expressed as a *linear function* of the elements of the sequence representations.

[14]The initial design of this project included training on aligned sentences. This plan has been rejected as no VAE architecture was found that would rely on parallel sentences and lead to convergent training. However, the collected parallel data was used as a bilingual, non-aligned corpus.

[15]https://paracrawl.eu/

[16]https://opus.nlpl.eu/

5.2.1. Linguistic acceptability. The linguistic acceptability task requires classifying sentences into two classes based on the feature whether they are grammatically and semantically well-formed or not. The models were trained to perform this task on the English CoLA dataset[17] by Warstadt, Singh & Bowman (2018) and its Hungarian version HuCoLA[18] created by Ligeti-Nagy et al. (2022).

5.2.2. Sentiment analysis. Binary sentiment analysis implies classifying sentences reflecting positive or negative opinions and feelings. SST2 is a binary classification version of the Stanford Sentiment Treebank released by Socher et al. (2013). In SST, sentiment labels are assigned to phrases, but sentence-level annotation is also available. The downloaded dataset[19] included sentences labelled with values between 0 and 1 (0 indicates an absolute negative sentiment, while 1 indicates an absolute positive one). These values where rounded to integer labels except the ones indicating a "neutral" sentiment from the interval (0.4, 0.6] (these were omitted). After this conversion, 6,919 training, 1,822 validation and 876 test data points remained.

The Hungarian Twitter Sentiment Corpus[20] consists of tweets classified into 5 sentiment classes. The corpus was converted into a binary classification dataset by removing sentences with label 3 (neutral) and mapping all other labels to 0 or 1. Finally, the data was split into train, validation and test sets with sizes of 2,193, 273 and 273 data points, respectively.

5.2.3. Linguistic probing tasks. The goal of the evaluation on the Tense and SOMO tasks was to study whether a sentence encoding contains information about the root verb tense, and semantic integrity of a sentence, respectively. Semantic integrity is understood here as a binary indicator of the fact whether a noun or verb in the sentence was randomly replaced with another word of the same part of speech. In Conneau, Kruszewski et al. (2018), these tasks are considered relevant mainly to the semantic evaluation of sentence embeddings, although Verb Tense is also connected to syntax. The linguistic probing tasks for English are available in SentEval[21], but they had to be created for Hungarian. This was done as follows: the 300K subset of the 2020 Hungarian news subcorpus was downloaded from the Leipzig Corpora Collection[22] by Goldhahn, Eckart & Quasthoff (2012). Morphological and syntactic dependency analysis were performed on these sentences using the e-magyar text processing system by Indig et al. (2019) and Váradi et al. (2018). This allowed to annotate the sentences as follows:

- If the root of the main clause was a verb in past tense, the sentence was labelled "1" and "0" if it was a verb in present or future tense. This contributed to the Tense probing task.
- A randomly picked noun or verb w_1 in the sentence was replaced with another noun or verb w_2 from the downloaded set of 300K sentences such that the morphological properties

[17]Accessed through https://huggingface.co/datasets/glue.

[18]https://huggingface.co/datasets/NYTK/HuCOLA

[19]https://github.com/frankaging/SST2-Sentence

[20]http://opendata.hu/dataset/hungarian-twitter-sentiment-corpus

[21]https://github.com/facebookresearch/SentEval/tree/main/data/probing

[22]urlhttps://wortschatz.uni-leipzig.de/en/download

Table 2. Linguistic probing examples with English translation. Bold text indicates the words on which the annotation is based

Task	Sentence	Label
Tense	Tettét állami kitüntetéssel **ismerték el.** *Her action **was recognized** with a state award.*	1
SOMO	A csapatok több kategóriában mérhetik össze gasztronómiai **szívritmusukat.** *The teams can compare their gastronomic **heart rate** in several categories.*	1

of w_1 and w_2 coincide. For each sentence that contained at least one verb or noun a pseudo-random number generator (random coin flipping) determined whether to perform this operation or not. If the operation was performed, the sentence was labelled "1" and "0" otherwise.

After annotating the sentences, 50K labelled sentences were randomly chosen for each linguistic probing task in such a way that the distribution of the labels should be balanced in each sample. For each task, a training split of 48K sentences as well as a validation and test split of 1,000 sentences each were created. As the source data of each probing task was the same, some of the sentences can occur in multiple probing tasks. Examples are given in Table 2.

6. TRAINING DETAILS

6.1. Pre-training

The pre-training procedure followed the main ideas outlined by Li et al. (2019): the model was first trained without applying the KL loss, then the regularization loss multiplier β was annealed. A KL threshold λ was also applied. However, the full pre-training plan was not predefined but altered dynamically based on the online monitoring of the loss curves. More specifically, the pre-training was split into the four phases described below:

1. β was set to zero. The reconstruction loss did not decrease further after iteration 60,000 and this phase was stopped. A cross entropy loss of 39.28 was reached.
2. β and λ were set to 0.05 and 0.3, respectively. This phase was shut down at iteration 40,000. Although the reconstruction loss did not stop decreasing at this point, the KL loss converged.
3. β was linearly annealed, with $\lambda = 0.2$. Although the overall training loss increased during this phase due to the growing β multiplier, the KL loss calculated with $\beta = 1$ decreased. As the reconstruction quality started to decrease at iteration 80,000, the phase was stopped. The final value of β was 0.26.
4. Training continued with a constant β of 0.26 and a λ of 0.2 until convergence. The last checkpoint was made at iteration 45,600. A final cross entropy loss of 14.15 and a KL loss of 193.65 (calculated with $\beta = 1$) were reached.

The AdamW optimizer with weight decay parameter set to 10^{-6} was applied. Linear cyclical learning rate scheduling, described by Smith (2015), was used in all phases. Batch sizes dynamically changed by bucketing according to sequence length.[23] Thus the maximal batch size was 256 and the minimal was 64.

With this configuration, less than a full epoch of the collected pre-training data was used. The model was trained on approximately 50M data points (containing both English and Hungarian sentences), which is less than $\frac{1}{3}$ of the full training corpus. It is likely that a training configuration could be found that allows incorporate more data and decrease the loss further, but finding such a configuration requires more experiments.

6.2. Fine-tuning

The fine-tuning setup is important as it determines how the models are compared. A serious complication is connected to setting the hyperparameters: it can seriously affect the training results, but the best configuration needs to be found for each model independently. Hyperparameter tuning sessions were run for each model-dataset pair with Bayesian search and the Hyperband early-stopping technique introduced by Li et al. (2016). The number of training epochs was fixed and set to 2 with checkpoints after both epochs. The AdamW optimizer and the one-cycle learning rate and momentum schedule proposed by Smith (2018) were applied in every case, but their parameters were tuned. When the fine-tuning updated all model parameters, 12 runs were allowed within every hyperparameter tuning session. When only the classifier head parameters were trained, a session could include only 8 runs, as in this case the initial learning rate was fixed at the value of 10^{-5} and the maximal and minimal momentum parameters were also predefined with values 0.95 and 0.85.

Fine-tuning batches were constructed by bucketing with a largest batch size of 64 examples. The exception is the Hungarian Twitter Sentiment corpus, which had the smallest training split among all the datasets. In order to enable more iterations, the maximal batches consisted of only 32 data points.

7. RESULTS

Fine-tuning results are reported in Tables 3–6. MCC stands for the Matthews Correlation Coefficient. It is a metric that takes values in the interval between -1 and 1. It is usually preferred to accuracy as it gives a more realistic evaluation of classifier models in the case when

Table 3. MCC scores on Hungarian data without full fine-tuning

Model	Tense	SOMO	HuCoLA	Sentiment
huBERT	**0.4367**	0.1309	**0.0517**	0.1629
XLM-R	0.0000	**0.1544**	0.0000	0.0000
BiVaSE	0.2902	0.0667	0.0027	**0.1682**

[23]This means that smaller batches included longer sequences.

Table 4. MCC scores on English data without full fine-tuning

Model	Tense	SOMO	CoLA	Sentiment
BERT	**0.6670**	0.0810	**0.0464***	−0.0192
XLM-R	0.5063	**0.1050**	0.0000*	0.0000
BiVaSE	0.2901	−0.0083	0.0362	**0.0449**

Table 5. MCC scores on Hungarian data with full fine-tuning

Model	Tense	SOMO	HuCoLA	Sentiment
huBERT	**0.9370**	**0.8023**	**0.6346**	**0.6720**
XLM-R	0.9265	0.6653	0.0000	0.5146
BiVaSE	0.8610	0.1706	0.1357	0.6023

Table 6. MCC scores on English data with full fine-tuning

Model	Tense	SOMO	CoLA	Sentiment
BERT	0.8269	**0.4255**	**0.5383***	0.1058
XLM-R	**0.8400**	0.3121	0.2621*	0.0000
BiVaSE	0.7000	0.0000	0.0000	**0.1137**

the label distribution is unbalanced. For a detailed analysis of the advantages of MCC over accuracy, please refer to Chicco & Jurman (2020).

Each cell in the tables below contains the score measured on the test set. The ecxeptions are scores annotated with a * in Tables 4 and 6: these are the best validation results. This is due to the fact the CoLA test split labels are not publicly available. Submissions can be made to an evaluation server[24] that returns the MCC score (there are some limitations, e.g. the number of submissions per day). Only the predictions made by BiVaSE were evaluated with this method.

As huBERT and BERT were trained on a large amount of monolingual data, they were expected to achieve the highest scores. XLM-R supports 100 languages and therefore it could be hypothesized that it can handle tasks specific to a certain language less effectively. These hypotheses concerning the non-variational Transformer-based models usually hold. The overall performance of BiVaSE is lower than that of the concurrent models, but it could achieve competitive results on several datasets.

Predicting the verb tense proved to be the simplest of all tasks. Tables 3–4 show that the models are often able to capture this kind of information without fine-tuning of the Transformer layers.

[24]The test set predictions can be submitted at https://www.kaggle.com/c/cola-out-of-domain-open-evaluation and https://www.kaggle.com/c/cola-in-domain-open-evaluation.

The only exception is XLM-R on the Hungarian data, but this model could also solve the task with high scores after full fine-tuning.

It is not obvious whether the pre-trained sentence representations learned by the models also contained any useful information relevant to other tasks. The MCC scores remained close to 0 (i.e. to "random guessing"), although they were almost always positive. The results given in Tables 5–6 indicate that even short full fine-tuning runs of 2 epochs could significantly enhance the performances. The update of the encoder parameters affected BiVaSE the least: the latent space regularization did not necessarily help the model adapt to specific tasks more quickly. However, it should be noted that full fine-tuning did not always guarantee the desirable convergence for the other models either. Some examples for this are sentiment analysis in English with BERT and XLM-R and predicting linguistic acceptability on Hungarian data with XLM-R. In the latter case, unbalanced label distribution[25] could explain the failure to generalize well.

Furthermore, the results indicate that full scale fine-tuning is inevitable to obtain high-quality results that may be required in real-world applications. It is not surprising that models with a few hundred million parameters are not large enough to be effectively used without adapting their layers to specific tasks. However, verb tense prediction can work relatively well even without tuning the pre-trained Transformer layers, which is most likely due to the fact verbs have a significant impact on the overall sentence structure and the tense is usually indicated by specific morphemes (i.e. this information can be captured by merely focusing on the surface form).

Another interesting observation is that huBERT tends to achieve better results after full fine-tuning than BERT. This is unexpected as both the training conditions and the datasets on which these models were trained are similar. The experiments presented in this paper are insufficient to provide an explanation for this observation.

8. CONCLUSIONS

It has been attempted to pre-train BiVaSE, a bilingual sentence encoder on unstructured text data with the VAE objective and fine-tune it on task-specific datasets. MCC scores indicate that this model has not reached the performance of other Transformer-based pre-trained models, but it has also been showed that it could learn to capture important linguistic information during pre-training (e.g. verb tense) and its performance can be increased through fine-tuning.

As far as the hypotheses formulated in Section 4.1 are concerned, the following conclusions can be drawn:

- **Latent space regularization supports classification.** The results outlined in Section 7 do not support this statement. There is no evidence that BiVaSE could benefit from latent space regularization.
- **A Transformer encoder can be trained from scratch using the VAE objective.** Tha fact that BiVaSE achieved competitive results in multiple cases indicates that it learned some general linguistic features during pre-training. Consequently, using the autoencoder or VAE objective can seriously be taken into consideration when choosing a pre-training method for a Transformer-based NLP model even in a multilingual setup.

[25]In fact, an MCC score of 0 can correspond to approximately 80% accuracy on HuCoLA.

In other words, the VAE objective (that implies using an autoencoder) should be regarded as a possible option for pre-training, but the latent space regularization does not necessarily enhance the performance of the resulting model.

Furthermore, it was observed that the linguistic feature that the models captured most easily was the verb tense. Every other task required fine-tuning to achieve results that are notably better than random guessing. It is interesting that distinguishing well-formed sentences from ill-formed ones generally proved to be a difficult task even after full fine-tuning. The most obvious explanation for this is the fact that the texts on which the models are pre-trained are datasets crawled from the web. These datasets contain much noise such as ill-formed sentences.

The introduction of BiVaSE is intended to be an initial step towards finding out whether traditional LM or MLM can be replaced with the VAE or simple autoencoder objective to pre-train general-purpose sentence encoders, including multilingual ones. In future work, more experiments need to be carried out which should focus on processing more data during pre-training and evaluating the effects of using monolingual, bilingual or massively multilingual data.

REFERENCES

Abdelali, Ahmed, Francisco Guzman, Hassan Sajjad and Stephan Vogel. 2014. The AMARA corpus: Building parallel language resources for the educational domain. Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14). 1856–1862.

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

Bari, M. Saiful, Batool Haider and Saab Mansour. 2021. Nearest neighbour few-shot learning for cross-lingual classification. CoRR. abs/2109.02221.

Bengio, Yoshua, Réjean Ducharme, Pascal Vincent and Christian Janvin. 2003. A neural probabilistic language model. Journal of Machine Learning Research 3. 1137–1155.

Bowman, Samuel R., Gabor Angeli, Christopher Potts and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. 632–642.

Bowman, Samuel R., Luke Vilnis, Oriol Vinyals, Andrew M. Dai, Rafal Józefowicz and Samy Bengio. 2016. Generating sentences from a continuous space. Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning (CoNLL). 10–21.

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

Cer, Daniel, 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.

Chicco, Davide and Giuseppe Jurman. 2020. The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation. BMC Genomics 21(1). 6.

Christodoulopoulos, Christos and Mark Steedman. 2014. A massively parallel corpus: The Bible in 100 languages. Language Resources and Evaluation 49. 1–21.

Chung, Junyoung, Caglar Gulcehre, Kyunghyun Cho and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. Proceedings of the NIPS 2014 Workshop on Deep Learning and Representation Learning.

Conneau, Alexis, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. CoRR. abs/1911.02116.

Conneau, Alexis and Douwe Kiela. 2018. Senteval: An evaluation toolkit for universal sentence representations. CoRR. abs/1803.05449.

Conneau, Alexis, Douwe Kiela, Holger Schwenk, Loïc Barrault and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing. 670–680.

Conneau, Alexis, Germán Kruszewski, Guillaume Lample, Loïc Barrault and Marco Baroni. 2018. What you can cram into a single vector: Probing sentence embeddings for linguistic properties. CoRR. abs/1805.01070.

Conneau, Alexis and Guillaume Lample. 2019. Cross-lingual language model pretraining. Advances in Neural Information Processing Systems 32.

Conneau, Alexis, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk and Veselin Stoyanov. 2018. XNLI: Evaluating cross-lingual sentence representations. Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. 2475–2485.

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

Eger, Steffen, Andreas Rücklé and Iryna Gurevych. 2019. Pitfalls in the evaluation of sentence embeddings. CoRR. abs/1906.01575.

Fan, Angela, Shruti Bhosale, Holger Schwenk, Zhiyi Ma, Ahmed El-Kishky, Siddharth Goyal, Mandeep Baines, Onur Celebi, Guillaume Wenzek, Vishrav Chaudhary, Naman Goyal, Tom Birch, Vitaliy Liptchinsky, Sergey Edunov, Edouard Grave, Michael Auli and Armand Joulin. 2020. Beyond English-centric multilingual machine translation. CoRR. abs/2010.11125.

Feng, Fangxiaoyu, Yinfei Yang, Daniel Cer, Naveen Arivazhagan and Wei Wang. 2020. Language-agnostic BERT sentence embedding. CoRR. abs/2007.01852.

Firth, John Rupert. 1957. A synopsis of linguistic theory 1930–55. In F.R. Palmer (ed.) Studies in linguistic analysis: Special volume of the Philological Society. Oxford: The Philological Society. 1–32.

Gage, Philip. 1994. A new algorithm for data compression. The C Users Journal archive 12. 23–38.

Goldhahn, Dirk, Thomas Eckart and Uwe Quasthoff. 2012. Building large monolingual dictionaries at the Leipzig Corpora Collection: From 100 to 200 languages. Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12). 759–765.

Harris, Zellig. 1954. Distributional structure. Word 10(2–3). 146–162.

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

Indig, Balázs, Bálint Sass, Eszter Simon, Iván Mittelholcz, Noémi Vadász and Márton Makrai. 2019. One format to rule them all – The emtsv pipeline for Hungarian. Proceedings of the 13th Linguistic Annotation Workshop. 155–165.

Kingma, Diederik P., Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever and Max Welling. 2016. Improved variational inference with inverse autoregressive flow. Proceedings of the 30th International Conference on Neural Information Processing Systems. 4743–4751.

Kingma, Diederik P. and Max Welling. 2014. Auto-encoding variational bayes. 2nd International Conference on Learning Representations, ICLR 2014.

Kiros, Ryan, Yukun Zhu, Ruslan Salakhutdinov, Richard S. Zemel, Raquel Urtasun, Antonio Torralba and Sanja Fidler. 2015. Skip-thought vectors. Advances in Neural Information Processing Systems 28. 3294–3302.

Li, Bohan, Junxian He, Graham Neubig, Taylor Berg-Kirkpatrick and Yiming Yang. 2019. A surprisingly effective fix for deep latent variable modeling of text. CoRR. abs/1909.00868.

Li, Chunyuan, Xiang Gao, Yuan Li, Xiujun Li, Baolin Peng, Yizhe Zhang and Jianfeng Gao. 2020. Optimus: Organizing sentences via pre-trained modeling of a latent space. Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP). 4678–4699.

Li, Lisha, Kevin G. Jamieson, Giulia DeSalvo, Afshin Rostamizadeh and Ameet Talwalkar. 2016. Efficient hyperparameter optimization and infinitely many armed bandits. CoRR. abs/1603.06560.

Ligeti-Nagy, Noémi, Gergő Ferenczi, Enikő Héja, Kinga Jelencsik-Mátyus, László János Laki, Noémi Vadász, Zijian Győző Yang and Tamás Váradi. 2022. HuLu: Magyar nyelvű benchmark adatbázis kiépítése a neurális nyelvmodellek kiértékelése céljából [HuLu: Building a Hungarian benchmark for evaluating neural language models]. XVIII. Magyar Számítógépes Nyelvészeti Konferencia. 431–446.

Lison, Pierre and Jörg Tiedemann. 2016. Opensubtitles2016: Extracting large parallel corpora from movie and TV subtitles. Proceedings of the 10th International Conference on Language Resources and Evaluation LREC 2016. 923–929.

Mel'chuk, Igor. 1974. Opyt teorii lingvisticheskikh modeley «Smysl – Tekst». [Outline of a theory of meaning–text linguistic models]. Moscow: Nauka.

Mickus, Timothee, Denis Paperno, Mathieu Constant and Kees van Deemter. 2019. What do you mean, BERT? Assessing BERT as a distributional semantics model. CoRR. abs/1911.05758.

Miháltz, Márton. 2013. OpinHuBank: szabadon hozzáférhető annotált korpusz magyar nyelvű véleményelemzéshez [OpinHuBank: a freely accessible annotated corpus for sentiment analysis in Hungarian]. IX. Magyar Számítógépes Nyelvészeti Konferencia. 343–345.

Mikolov, Tomas, Kai Chen, Greg Corrado and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. CoRR. abs/1301.3781.

Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg S. Corrado and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. Advances in Neural Information Processing Systems 26.

Nemeskey, Dávid Márk. 2020. Natural language processing methods for language modeling. Doctoral dissertation. Eötvös Loránd University, Budapest.

Nemeskey, Dávid Márk. 2021. Introducing huBERT. XVII. Magyar Számítógépes Nyelvészeti Konferencia (MSZNY2021). 3–14.

Perone, Christian S., Roberto Pereira Silveira and Thomas S. Paula. 2018. Evaluation of sentence embeddings in downstream and linguistic probing tasks. CoRR. abs/1806.06259.

Radford, Alec and Karthik Narasimhan. 2018. Improving language understanding by generative pre-training. arXiv.

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

Ramírez-Sánchez, Gema, Jaume Zaragoza-Bernabeu, Marta Bañón and Sergio Ortiz-Rojas. 2020. Bifixer and bicleaner: Two open-source tools to clean your parallel data. Proceedings of the 22nd Annual Conference of the European Association for Machine Translation. 291–298.

Reimers, Nils and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using siamese BERT-networks. Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing.

Reimers, Nils and Iryna Gurevych. 2020. Making monolingual sentence embeddings multilingual using knowledge distillation. CoRR. abs/2004.09813.

Rezende, Danilo Jimenez, Shakir Mohamed and Daan Wierstra. 2014. Stochastic backpropagation and approximate inference in deep generative models. In E.P. Xing and T. Jebara (eds.) Proceedings of Machine Learning Research, Vol. 32: Proceedings of the 31st International Conference on Machine Learning. Bejing: PMLR. 1278–1286.

Rozis, Roberts and Raivis Skadiņš. 2017. Tilde MODEL – multilingual open data for EU languages. Proceedings of the 21st Nordic Conference on Computational Linguistics. 263–265.

Rücklé, Andreas, Steffen Eger, Maxime Peyrard and Iryna Gurevych. 2018. Concatenated p-mean word embeddings as universal cross-lingual sentence representations. CoRR. abs/1803.01400.

Sánchez-Cartagena, Víctor M., Marta Bañón, Sergio Ortiz-Rojas and Gema Ramírez-Sánchez. 2018. Prompsit's submission to WMT 2018 Parallel Corpus Filtering shared task. Proceedings of the Third Conference on Machine Translation, Vol. 2: Shared Task Papers. 955–962.

Schwenk, Holger, Vishrav Chaudhary, Shuo Sun, Hongyu Gong and Francisco Guzmán. 2019. Wikimatrix: Mining 135m parallel sentences in 1620 language pairs from Wikipedia. CoRR. abs/1907.05791.

Schwenk, Holger, Guillaume Wenzek, Sergey Edunov, Edouard Grave and Armand Joulin. 2019. Ccmatrix: Mining billions of high-quality parallel sentences on the WEB. CoRR. abs/1911.04944.

Smith, Leslie N. 2015. No more pesky learning rate guessing games. CoRR. abs/1506.01186.

Smith, Leslie N. 2018. A disciplined approach to neural network hyper-parameters: Part 1 – learning rate, batch size, momentum, and weight decay. CoRR. abs/1803.09820.

Socher, Richard, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, 1631–1642.

Tiedemann, Jörg. 2012. Parallel data, tools and interfaces in OPUS. Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12). 2214–2218.

Varga, Dániel, László Németh, Péter Halácsy, András Kornai, Viktor Trón and Viktor Nagy. 2005. Parallel corpora for medium density languages. Proceedings of the Ranlp 2005. 590–596.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser and Illia Polosukhin. 2017. Attention is all you need. CoRR. abs/1706.03762.

Váradi, Tamás, Eszter Simon, Bálint Sass, Iván Mittelholcz, Attila Novák and Balázs Indig. 2018. E-magyar – A digital language processing system. Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018). 1307–1312.

Wang, Alex, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP. 353–355.

Warstadt, Alex, Amanpreet Singh and Samuel R. Bowman. 2018. Neural network acceptability judgments. CoRR. abs/1805.12471.

Williams, Philip and Barry Haddow. 2021. The ELITR ECA corpus. CoRR. abs/2109.07351.

Wołk, Krzysztof and Krzysztof Marasek. 2014. Building subject-aligned comparable corpora and mining it for truly parallel sentence pairs. Procedia Technology 18. 126–132.

Zhang, Biao, Deyi Xiong, Jinsong Su, Hong Duan and Min Zhang. 2016. Variational neural machine translation. Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. 521–530.

Appendix

Opus Corpora

The full list of Opus corpora from which the pre-training data for BiVASE was constructed is given below:

- **WikiMatrix v1**:[26] parallel corpora from Wikimedia by Schwenk, Chaudhary et al. (2019)
- **CCMatrix v1**:[27] parallel sentences from web crawls by Schwenk, Wenzek et al. (2019) and Fan et al. (2020)
- **wikimedia v20210402**:[28] Wikipedia translations
- **TildeMODEL v2018**:[29] multilingual open data corpus by Rozis & Skadiņš (2017)
- **DGT v2019**:[30] translation memories provided by the JRC.
- **JRC-Acquis v3.0**:[31] legislative texts of the European Union
- **EUbookshop v2**:[32] https://opus.nlpl.eu/EUbookshop-v2.php documents from the EU bookshop
- **Europarl v8**:[33] a parallel corpus from the European Parliament web
- **ELITR-ECA v1**:[34] by Williams & Haddow (2021) a multilingual corpus from documents published by the European Court of Auditors
- **TED2020 v1**:[35] described by Reimers & Gurevych (2020) a crawl of TED and TED-X transcripts
- **QED v2.0a**:[36] by Abdelali et al. (2014) subtitles for educational videos and lectures
- **EMEA v3**:[37] a parallel corpus that contains documents from the European Medicines Agency
- **GNOME v1**:[38] a corpus of GNOME localization files
- **Books v1**:[39] copyright free books aligned by Andras Farkas[40]
- **ECB v1**:[41] website and documentation from the European Central Bank
- **bible-uedin v1**:[42] a collection of translations of the Bible by Christodoulopoulos & Steedman (2014)

[26]https://opus.nlpl.eu/WikiMatrix-v1.php

[27]https://opus.nlpl.eu/CCMatrix-v1.php

[28]https://opus.nlpl.eu/wikimedia-v20210402.php

[29]https://opus.nlpl.eu/TildeMODEL-v2018.php

[30]https://opus.nlpl.eu/DGT-v2019.php

[31]https://opus.nlpl.eu/JRC-Acquis-v3.0.php

[32]https://opus.nlpl.eu/EUbookshop-v2.php

[33]https://opus.nlpl.eu/Europarl-v8.php

[34]https://opus.nlpl.eu/ELITR-ECA-v1.php

[35]https://opus.nlpl.eu/TED2020-v1.php

[36]https://opus.nlpl.eu/QED-v2.0a.php

[37]https://opus.nlpl.eu/EMEA-v3.php

[38]https://opus.nlpl.eu/GNOME-v1.php

[39]https://opus.nlpl.eu/Books-v1.php

[40]https://farkastranslations.com/bilingual_books.php

[41]https://opus.nlpl.eu/ECB-v1.php

[42]https://opus.nlpl.eu/bible-uedin-v1.php

- **Wikipedia v1.0**:[43] parallel sentences extracted from Wikipedia by Wołk & Marasek (2014)
- **KDE4 v2**:[44] a corpus of KDE4 localization files
- **Tatoeba v2021-07-22**:[45] a collection of translated sentences from Tatoeba[46]
- **PHP v1**:[47] a parallel corpus originally extracted from the PHP documentation
- **GlobalVoices v2018q4**:[48] a parallel corpus of news stories
- **EUconst v1**:[49] a parallel corpus from the European Constitution
- **ELRC-3382 v1**:[50] COVID-19 EU presscorner v1 dataset.
- **WMT-News v2019**:[51] a parallel corpus of News Test Sets provided by WMT
- **ELRC-2922 v1**:[52] COVID-19-HEALTH Wikipedia dataset
- **ELRC-2923 v1**:[53] COVID-19 EUROPARL dataset
- **Ubuntu v14.10**:[54] a corpus of Ubuntu localization files
- **OpenSubtitles v2018**:[55] a collection of translated movie subtitles[56] by Lison & Tiedemann (2016)

[43]https://opus.nlpl.eu/Wikipedia-v1.0.php

[44]https://opus.nlpl.eu/KDE4-v2.php

[45]https://opus.nlpl.eu/Tatoeba-v2022-03-03.php

[46]https://tatoeba.org/en/

[47]https://opus.nlpl.eu/PHP-v1.php

[48]https://opus.nlpl.eu/GlobalVoices-v2018q4.php

[49]https://opus.nlpl.eu/EUconst-v1.php

[50]https://opus.nlpl.eu/ELRC_3382-v1.php

[51]https://opus.nlpl.eu/WMT-News-v2019.php

[52]https://opus.nlpl.eu/ELRC_2922-v1.php

[53]https://opus.nlpl.eu/ELRC_2923-v1.php

[54]https://opus.nlpl.eu/Ubuntu-v14.10.php

[55]https://opus.nlpl.eu/OpenSubtitles-v2018.php

[56]http://www.opensubtitles.org/