“© 2022 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.”

Regressing Word and Sentence Embeddings for Low-Resource Neural Machine Translation

Inigo Jauregi Unanue, Ehsan Zare Borzeshi, and Massimo Piccardi

***Abstract*—In recent years, neural machine translation (NMT) has achieved unprecedented performance in automated translation of resource-rich languages. However, it has not yet managed to achieve a comparable performance over the many low-resource languages and specialized translation domains, mainly due its tendency to overfit small training sets and consequently strive on new data. For this reason, in this paper we propose a novel approach to regularize the training of NMT models to improve their performance over low-resource language pairs. In the proposed approach, the model is trained to co-predict the target training sentences both as the usual categorical outputs (i.e., sequences of words) and as word and sentence embeddings. The fact that word and sentence embeddings are pre-trained over large corpora of monolingual data helps the model generalize beyond the available translation training set. Extensive experiments over three low-resource language pairs have shown that the proposed approach has been able to outperform strong state-of-the-art baseline models, with more marked improvements over the smaller training sets (e.g., up to $+6.57$ BLEU points in Basque-English translation). A further experiment on unsupervised NMT has also shown that the proposed approach has been able to improve the quality of machine translation even with no parallel data at all.**

***Impact Statement*—Neural machine translation (NMT) is the contemporary state of the art for machine translation. However, NMT models are notoriously data-hungry and typically require very large datasets to be trained effectively, in the order of millions of parallel sentences from the source and target languages. For the remaining language pairs, which are collectively classified as "low-resource", NMT still struggles to achieve a comparable level of performance. For this reason, in this paper we propose a novel training approach for NMT models that leverage existing, pre-trained word and sentence embeddings to improve the models' performance when only limited parallel training data are available. The experimental results over three low-resource language pairs show that the proposed approach has been able to improve the models' performance by several percentage points in the majority of cases. We also show that the proposed approach has been able to improve the models' performance in unsupervised NMT, where no parallel data are needed at all. We envisage that the proposed approach – aptly nicknamed *ReWE+ReSE* – could be easily integrated in mainstream deep learning libraries and achieve significant adoption. To facilitate the uptake, we release all our code and training set-ups publicly.**

***Index Terms*—Machine translation, neural machine translation, regularization, sentence embeddings, word embeddings.**

This manuscript was submitted for review the x of July of 2021.

I. Jauregi Unanue is with University of Technology Sydney, Sydney, NSW 2007, Australia, and RoZetta Technology, Sydney, NSW 2000, Australia (e-mail: inigo.jauregi@rozettatechnology.com).

E. Zare Borzeshi is with Microsoft Commercial Software Engineering (CSE), Zürich, Switzerland (e-mail: ehsan.zareborzeshi@microsoft.com).

M. Piccardi is with University of Technology Sydney, Sydney, NSW 2007, Australia (e-mail: massimo.piccardi@uts.edu.au).

I. Introduction

Machine translation (MT) is a field of natural language processing (NLP) focussing on the automatic translation of sentences from a *source* language to a *target* language. In recent years, the field has been progressing rapidly mainly thanks to the advances in deep learning and the advent of neural machine translation (NMT). The first NMT model was presented in 2014 by Sutskever et al. [1] and consisted of a plain *encoder-decoder* architecture based on recurrent neural networks (RNNs). In the following years, a series of improvements has led to major performance increases, including the attention mechanism (a word-alignment model between words in the source and target sentences) [2], [3] and the transformer (a non-recurrent neural network that offers a highly-parallelizable alternative to RNNs) [4]. As a result, NMT models have significantly outperformed traditional approaches such as phrase-based statistical machine translation (PBSMT) [5] in many translation contests (e.g., the WMT conference series). Nowadays, the majority of MT systems in use utilise NMT in some form.

However, NMT models are not exempt from limitations. The main is their tendency to overfit the training set due to their typically massive number of parameters. The direct consequence of overfitting is an inadequate and often unfluent performance of the models once deployed in field. While this issue can be countered with training sets of increasingly large size, such as those available for resource-rich language pairs (e.g., French-English, English-Chinese), it remains a very challenging problem for translation between many of the approximately $6,900$ languages currently used in the world [6]. In the context of contemporary machine translation, it is not uncommon for a resource-rich language pair to avail of parallel datasets of 1-10M+ sentences. Conversely, any translation dataset with only a few tens or hundreds of thousands parallel sentences can be currently classified as low-resource.

In technical terms, the main acknowledged cause of overfitting lies in the way NMT models are trained [7]. Usually, NMT models are trained with maximum likelihood estimation (MLE, or cross entropy) using a single reference translation in the target language for every example in the source language. For every training sentence pair, the MLE objective is to assign all of the model's probability to the reference target sentence, and zero to any alternative. However, a model could legitimately produce a translation that is different from the reference and is still perfectly correct (e.g., using paraphrases and synonyms). In addition, translations that deviate from the reference are not all equally incorrect. For instance, if the reference sentence contains the word *pigeon*, a prediction such as *bird*, even if only partially faithful, should score better than predicting a word such as *car*. Standard MLE is not able to leverage these important distinctions since it treats every word other than those in the provided reference as completely incorrect. In principle, MLE could achieve optimal performance with infinite training data, but in practice this is impossible to pursue since the available resources are inevitably limited. In particular, when the training data are scarce such as for low-resource language pairs or specialized domains, NMT models display a modest performance, and more traditional approaches (e.g., PBSMT [8]) often obtain better accuracies. As such, generalization of NMT systems still calls for significant improvement.

In our recent work [9], we have proposed a novel regularization technique for the training objective that is based on co-predicting words and their embeddings ("regressing word embeddings", or ReWE for short). ReWE is a module added to the decoder of a sequence-to-sequence model during training, so that the model is trained to jointly predict the next word in the translation (a categorical value) and its pre-trained word embedding (a multi-dimensional, continuous value). This approach has proved able to leverage the contextual information embedded in pre-trained word vectors, in particular with low/medium size training sets [9]. Given the increasing attention recently raised by *sentence embeddings* [10], [11] (inferred vectors that embed whole sentences), in this paper we extend this idea to the regression of sentence embeddings (ReSE). For every input sentence, ReSE uses a self-attention mechanism to infer a single, fixed-dimensional vector in output. During training, the model is trained to regress this inferred vector toward the pre-trained sentence embedding of the reference sentence. In specific, we propose jointly regressing word and sentence embeddings as a combined training regularizer, and we nickname the proposed approach as *ReWE+ReSE*. Overall, the main contributions of our paper are:

- The proposal of a new regularization technique for NMT training based on sentence embeddings (ReSE), and its joint use with a word-level regularizer (ReWE+ReSE).
- Extensive experiments over four language pairs: three low-resource (< 250K examples) datasets, and one high-resource (~ 5M examples) dataset. We show that using ReWE and ReSE jointly on the low-resource datasets can outperform strong state-of-the-art baselines based on transformers and long short-term memory networks (LSTMs). On the high-resource dataset, we show how the need for regularization decreases as the available training data increase.
- Insights on how ReWE and ReSE help improve the NMT models. Our analysis shows that the proposed regularizers make the decoder's output space more uniform, facilitating correct word classification.
- A further experiment on *unsupervised* machine translation, showing that the proposed regularizers are able to improve the quality of the translations even in the complete absence of parallel training data.

The rest of this paper is organized as follows: Section II presents and discusses the related work. Section III describes the models used as baselines. Section IV first recaps ReWE, and then presents the new regularizer, ReSE. Section V describes the experiments and discusses the results. Finally, Section VI concludes the paper.

II. Related Work

The related work is organized over the three main research subareas that have motivated this work: *regularization of NMT models*, *word and sentence embeddings* and *unsupervised NMT*.

A. Regularization of NMT Models

In recent years, the research community has dedicated much attention to the problem of overfitting in deep neural models. Several regularization approaches have been proposed in turn such as dropout [12], [13], data augmentation [14] and multi-task learning [15], [16]. Their common aim is to encourage the model to learn parameters that allow for better generalization.

In NMT, too, mitigating overfitting due to small training sets has been the focus of much research. Fadee et al. [14] have proposed augmenting the training data with synthetically-generated sentence pairs containing rare words. In this way, during training the model is able to see such rare words in a plausible context. Similarly, monolingual data in the target language have been used for data augmentation using "back-translation" [17], [18]. Basically, an existing target-to-source MT model is employed to translate the monolingual target data back to the source language, thus creating additional, quasi-parallel data that can be used to augment the available training set. However, the effectiveness of back-translation heavily depends on the quality of the auxiliary MT model, which is inherently limited in the case of low-resource language pairs. In [19], Kudo has proposed using multiple subword segmentations to improve the model's robustness, achieving notable improvements with low-resource languages and out-of-domain settings. Another line of work has focused on "smoothing" the output probability distribution over the target vocabulary [7], [20]. These approaches use token-level and sentence-level reward functions that push the model to spread the output probability over words other than the ground-truth reference. In a similar vein, Ma *et al.* in [21] have augmented the training objective with a bag-of-words representation of the reference sentence, assuming that all acceptable translations would share comparable bag-of-words. Finally, curriculum learning has also been explored to use the available training data more effectively [22], [23], [24]. In curriculum learning, the training samples are carefully selected so that the "easier" ones are used in the initial stages of training, while increasingly "difficult" samples are introduced as training progresses. Empirically, curriculum learning has proved effective at mollifying overfitting [23], [24].

B. Word and Sentence Embeddings

Recently, large pre-trained language models (LMs) such as ELMo [25], BERT [26], GPT [27] and T5 [28] have been

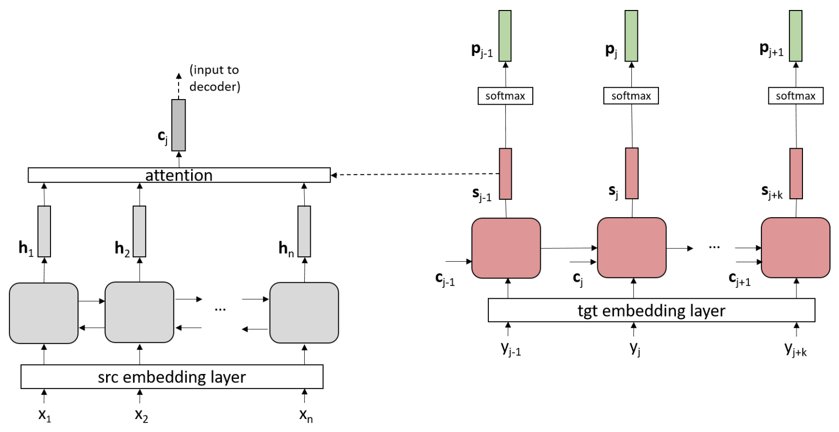


Fig. 1: Baseline NMT model. (Left) The encoder receives the input sentence and generates a context vector $\mathbf{c}_j$ for each decoding step using an attention mechanism. (Right) The decoder generates one-by-one the output vectors $\mathbf{p}_j$, which represent the probability distribution over the target vocabulary. During training $\mathbf{y}_j$ is a token from the ground truth sentence, but during inference the model uses its own predictions.

used to improve the performance of deep learning models in several NLP tasks (an approach often referred to as “transfer learning”). These pre-trained models are able to generate “contextual” embeddings for each word in a given sentence, supporting better disambiguation than conventional, “static” word embeddings [29], [30], and making for informative input features for downstream tasks. In addition to word embeddings, pre-trained LMs have been used to generate embeddings for larger chunks of texts (e.g., sentences, documents), which can also be used to improve downstream tasks [10], [31], [11].

In NMT models, word embeddings play an important role as input of both the encoder and the decoder. A recent paper has shown that contextual word embeddings provide effective input features for both stages [32]. However, very little research has been devoted to using word embeddings as *targets*. Kumar and Tsvetkov in [33] have removed the typical output softmax layer, forcing the decoder to generate continuous outputs. At inference time, they use a nearest-neighbour search in the word embedding space to select the word to predict. Their model allows for significantly faster training while performing on par with state-of-the-art models. Our approach differs from [33] in that our decoder generates continuous outputs *in parallel* with the standard softmax layer, and only during training to provide regularization. At inference time, the continuous output is ignored and prediction operates as in a standard NMT model. In this work, in addition to using word embeddings, we also explore the use of sentence embeddings generated with pre-trained LMs for NMT regularization. To the best of our knowledge, our model is the first to use embeddings as targets for regularization, and at both word and sentence level.

C. Unsupervised NMT

In extreme cases, a language pair may completely lack a parallel training set. In such cases, a possible approach is to use *unsupervised NMT* [34], [35], [36] which does not require any aligned, bilingual text for training and only learns to translate from monolingual text in both languages. Even though the accuracy of unsupervised NMT models is still much lower than that of their supervised counterparts, they have started to reach interesting levels. In addition, unsupervised NMT models can be the basis for few-shot NMT adaptation [37]. The architecture of unsupervised NMT systems differs from that of supervised systems in that it combines translation in both directions (source-to-target and target-to-source). Typically, a single encoder is used to encode sentences from both languages, and a separate decoder generates the translations in each language. The training of such systems follows three stages: 1) building a bilingual dictionary and word embedding space, 2) training two monolingual language models as denoising autoencoders [38], and 3) converting the unsupervised problem into a weakly-supervised one by use of back-translations [17]. For more details of unsupervised NMT models, we refer the reader to the original papers [34], [35], [36].

In this paper, we explore using the proposed regularization approach also for unsupervised NMT. Unsupervised NMT models still require substantial amounts of monolingual data for training, and often such amounts are not available. Therefore, these models, too, are expected to benefit from improved regularization.

III. The Baseline NMT model

In this section, we describe the NMT model that has been used as the basis for the proposed regularizer. It is a neural encoder-decoder architecture with attention [2] that can be regarded as a strong baseline as it incorporates both LSTMs and transformers as modules. Although we refer to this model simply as “baseline” in the rest of our paper, it holds state-of-the-art accuracy on several machine translation datasets [39], [40], [41]. Therefore the improvements over this baseline that we present in Section V set a new state-of-the-art accuracy. Let us note a source sentence with n tokens as $\mathbf{x} = \{x_1 \dots x_n\}$, and the corresponding target sentence with m tokens as $\mathbf{y} = \{y_1 \dots y_m\}$. First, the words in the source sentence are encoded by an embedding layer into respective

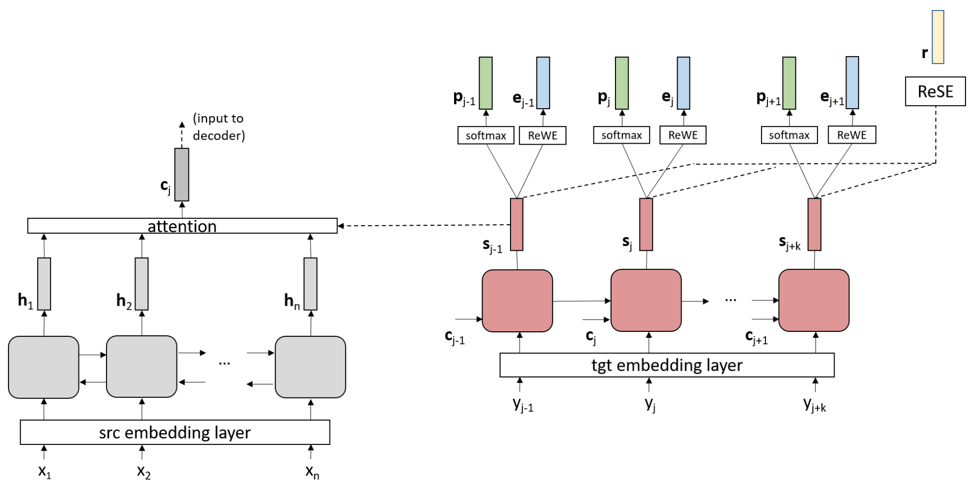


Fig. 2: Full model: Baseline + ReWE + ReSE. (Left) The encoder with the attention mechanism generates vectors $\mathbf{c}_j$ in the same way as the baseline system. (Right) The decoder generates one-by-one the output vectors $\mathbf{p}_j$, which represent the probability distribution over the target vocabulary, and $\mathbf{e}_j$, which is a continuous word vector. Additionally, the model can also generate another continuous vector, $\mathbf{r}$, which represents the sentence embedding.

word embeddings, $\mathbf{x}^e_1 \dots \mathbf{x}^e_n$:

$$\mathbf{x}^e_i = src\_emb(x_i) \quad i = 1 \dots n. \tag{1}$$

Then, the source sentence is encoded by a sequential module into its hidden vectors, $\mathbf{h}_1 \dots \mathbf{h}_n$:

$$\mathbf{h}_i = enc(\mathbf{h}_{i-1}, \mathbf{x}^e_i) \quad i = 1 \dots n \tag{2}$$

Next, for each decoding step $j = 1 \dots m$, an attention network provides a context vector $\mathbf{c}_j$ as a weighted average of all the encoded vectors, $\mathbf{h}_1 \dots \mathbf{h}_n$, conditional on the decoder's hidden vector at the previous step, $\mathbf{s}_{j-1}$:

$$\mathbf{c}_j = attn(\mathbf{h}_1 \dots \mathbf{h}_n, \mathbf{s}_{j-1}) \quad j = 1 \dots m \tag{3}$$

For this network, we have used the standard attention mechanism of Bahdanau *et al.* [2]. Given the context vector, $\mathbf{c}_j$, the decoder output at the previous step, $\mathbf{s}_{j-1}$, and the word embedding of the previous word in the target sentence, $\mathbf{y}^e_j$ (Eq. 4), the decoder generates vector $\mathbf{s}_j$ (Eq. 5). This vector is later transformed into a larger vector of the same size as the target vocabulary via learned parameters $\mathbf{W}$, $\mathbf{b}$ and a softmax layer (Eq. 6). The resulting vector, $\mathbf{p}_j$, is the inferred probability distribution over the target vocabulary at decoding step j. Fig. 1 depicts the full architecture of the baseline model.

$$\mathbf{y}^e_j = tgt\_emb(y_j) \quad j = 1 \dots m \tag{4}$$

$$\mathbf{s}_j = dec(\mathbf{c}_j, \mathbf{s}_{j-1}, \mathbf{y}^e_{j-1}) \quad j = 1 \dots m \tag{5}$$

$$\mathbf{p}_j = \text{softmax}(\mathbf{W}\mathbf{s}_j + \mathbf{b}) \tag{6}$$

The model is trained by minimizing the negative log-likelihood (NLL) which can be expressed as:

$$\mathcal{L}_{NLL} = -\sum_{j=1}^{m} \log \mathbf{p}_j(y_j) \tag{7}$$

where the probability of ground-truth word y_j has been noted as $\mathbf{p}_j(y_j)$. Minimizing the NLL is equivalent to MLE and results in assigning maximum probability to the words in the reference translation, $y_j, j = 1 \dots m$. The training objective is minimized with standard backpropagation over the training data, and at inference time the model uses beam search for decoding.

IV. Regressing word and sentence embeddings

As mentioned in the introduction, MLE suffers from some limitations when training a neural machine translation system. To alleviate these shortcomings, in our recent paper [9] we have proposed a new regularization technique based on regressing word embeddings (ReWE). In this section, we briefly review ReWE, and then present its extension to sentence embeddings (ReSE).

A. ReWE

Pre-trained word embeddings are trained on large monolingual corpora by measuring the co-occurences of words in text windows ("contexts"). Words that occur in similar contexts are assumed to have similar meaning, and hence, similar vectors in the embedding space. Our goal with ReWE is to incorporate the information embedded in the word vector in the loss function to encourage model regularization.

In order to generate continuous vector representations as outputs, we have added a ReWE block to the NMT baseline (Fig. 2). At each decoding step, the ReWE block receives the hidden vector from the decoder, $\mathbf{s}_j$, as input, and outputs another vector, $\mathbf{e}_j$, of the same size of the pre-trained word embeddings:

$$\begin{aligned} \mathbf{e}_j &= ReWE(\mathbf{s}_j) \\ &= \mathbf{W}_2(ReLU(\mathbf{W}_1\mathbf{s}_j + \mathbf{b}_1)) + \mathbf{b}_2 \end{aligned} \tag{8}$$

In Eq. (8), $\mathbf{W}_1$, $\mathbf{W}_2$, $\mathbf{b}_1$ and $\mathbf{b}_2$ are the learnable parameters of a two-layer feed-forward network with a Rectified Linear Unit (ReLU) as activation function between the layers. Vector $\mathbf{e}_j$ aims to reproduce the word embedding of the target word, and thus the distributional properties (or co-occurrences) of its contexts.

During training, the model is guided to regress the predicted vector, $\mathbf{e}_j$, toward the word embedding of the ground-truth word, $\mathbf{y}^e_j$. This is achieved by using a loss function that computes the distance between $\mathbf{e}_j$ and $\mathbf{y}^e_j$ (Eq. 9). Previous work [9] has shown that the cosine distance is empirically an effective distance between word embeddings and has thus been adopted as loss. This loss and the original NLL loss are combined together with a tunable hyperparameter, λ (Eq. 10). Therefore, the model is trained to jointly predict both a categorical and a continuous representation of the words. Even though the system is performing a single task, this setting could also be interpreted as a form of multi-task learning with different representations of the same targets.

$$\mathcal{L}_{ReWE} = \sum_{j=1}^{m}(1 - \cos(\mathbf{e}_j, \mathbf{y}^e_j)) \tag{9}$$

$$\mathcal{L}_w = \mathcal{L}_{NLL} + \lambda\mathcal{L}_{ReWE} \tag{10}$$

The word vectors of both the source ($\mathbf{x}^e$) and target ($\mathbf{y}^e$) vocabularies are initialized with pre-trained embeddings, and updated during training. At inference time, we ignore the outputs of the ReWE block and perform translation using only the categorical prediction.

B. Extending ReWE to sentence embeddings: ReSE

Sentence vectors, too, are extensively used as input representations in many NLP tasks such as text classification, paraphrase detection, natural language inference and question answering. The intuition behind them is very similar to that of word embeddings: sentences with similar meanings are expected to be close to each other in vector space. Several off-the-shelf sentence embedders are currently available and they can be easily integrated in deep learning models. Based on similar assumptions to the case of word embeddings, we hypothesize that an NMT model could also benefit from a regularization term based on regressing sentence embeddings (the ReSE block in Fig. 2).

The main difference of ReSE compared to ReWE is that it predicts a single regressed vector per sentence rather than one per word. Thus, ReSE first uses a self-attention mechanism to learn a weighted average of the decoder's hidden vectors, $\mathbf{s}_1 \dots \mathbf{s}_m$:

$$self\_attn(\mathbf{s}_1 \dots \mathbf{s}_m) = \sum_{j=0}^{m} \alpha_j \mathbf{s}_j \tag{11}$$

$$\alpha_j = \frac{e^{l_j}}{\sum_{k=0}^{m} e^{l_k}} \tag{12}$$

$$l_j = \mathbf{U}_2 \tanh(\mathbf{U}_1 \mathbf{s}_j) \tag{13}$$

where the α_j attention weights are obtained from Eqs. 12 and 13, and $\mathbf{U}_1$ and $\mathbf{U}_2$ are learnable parameters. Then, a two-layered neural network similar to ReWE's predicts the sentence vector, $\mathbf{r}$ (Eq. 14). Parameters $\mathbf{W}_3$, $\mathbf{W}_4$, $\mathbf{b}_3$ and $\mathbf{b}_4$ are also learned during training.

$$\begin{aligned} \mathbf{r} &= ReSE([\mathbf{s}_1, \dots, \mathbf{s}_m]) \\ &= \mathbf{W}_3(ReLU(\mathbf{W}_4\ self\_attn(\mathbf{s}_1 \dots \mathbf{s}_m) + \mathbf{b}_3)) + \mathbf{b}_4 \end{aligned} \tag{14}$$

Similarly to ReWE, a loss function computes the cosine distance between the predicted sentence vector, $\mathbf{r}$, and the sentence vector inferred with an external sentence embedder, $\mathbf{y}^r$:

$$\mathcal{L}_{ReSE} = 1 - \cos(\mathbf{r}, \mathbf{y}^r) \tag{15}$$

To embed the sentences, we have experimented with four different embedders:

- **avgEmbs**: A sentence embedding formed by the average of the word embeddings of the ground-truth sentence.
- **maxpoolEmbs**: A 1D max-pooling of the word embeddings of the ground-truth sentence.
- **USE**: The universal sentence encoder [10], a transformer-based network trained in a multi-task framework to generate versatile sentence embeddings.
- **SBERT**: Sentence-BERT [11], another transformer-based network trained in an unsupervised manner with a masked language model objective, and fine-tuned over a sentence similarity task.

The ReSE loss is finally added to the objective with an additional, tunable hyperparameter, β:

$$\mathcal{L}_{ws} = \mathcal{L}_{NLL} + \lambda\mathcal{L}_{ReWE} + \beta\mathcal{L}_{ReSE} \tag{16}$$

Since the number of sentences is significantly lower than that of the words, β typically needs to be higher than λ. Nevertheless, we tune it blindly using the validation set. At inference time, the model ignores the predicted word and sentence vectors and solely relies on the categorical prediction.

V. Experiments

We have carried out an ample range of experiments to probe the performance of the proposed regularization approaches. This section describes the datasets, the models and the hyper-parameters used, and presents and discusses all results.

A. Datasets

Four different language pairs have been selected for the experiments. The datasets for three language pairs have each less than 250K parallel sentences and can be regarded as low-resource. The last language pair has over 5M parallel sentences and allows us to explore the trade-off between the training set size and the effectiveness of the proposed regularizers.

- **En-Fr**: The English-French dataset (En-Fr) has been sourced from the IWSLT 2016 translation shared task[1]. This corpus contains translations of TED talks of very diverse topics. The training data provided by the organizers consist of $219,777$ translations which makes this dataset a low/medium-resource case. Following Denkowski and Neubig [42], the validation set has been formed by merging the 2013 and 2014 test sets from the same shared task, and the test set has been formed with the 2015 and 2016 test sets.
- **Cs-En**: The Czech-English dataset (Cs-En) is also from the IWSLT 2016 TED talks translation task. However, this dataset is approximately half the size of En-Fr as its training set consists of $114,243$ sentence pairs. Again following Denkowski and Neubig [42]), the validation set has been formed by merging the 2012 and 2013 test sets, and the test set by merging the 2015 and 2016 test sets. We rate this dataset as low-resource.
- **Eu-En**: The Basque-English dataset (Eu-En) has been collected from the WMT16 IT-domain translation shared task[2]. This is the smallest dataset, with only $89,413$ sentence pairs in the training set. However, only $2,000$ sentences in the training set have been translated by human annotators. The remaining sentence pairs are translations of IT-domain short phrases and Wikipedia titles. Therefore, this dataset should be rated as very low-resource. For this dataset, we have used the validation and test sets ($1,000$ sentences each) provided in the shared task.
- **De-En**: The German-English dataset (De-En) has been taken from the WMT18 news translation shared task[3]. The training set contains over 5M sentence pairs collected from the *Europarl*, *CommonCrawl* and *Newscommentary* parallel corpora. As validation and test sets, we have used the *newstest2017* and the *newstest2018* datasets, respectively. This dataset is high-resource by contemporary standards.

All the datasets have been pre-processed with the Moses tokenizer[4]. Additionally, words have been split into subword units using byte pair encoding (BPE) [43]. For the BPE merge operations parameter, we have used $32,000$ (the default value) for all the datasets, except for Eu-En where we have set it to $8,000$ since this dataset is much smaller. Experiments have been performed at both word and subword level since morphologically-rich languages such as German, Czech and Basque can benefit greatly from translating at subword level.

[1]IWSLT16: https://workshop2016.iwslt.org/

[2]WMT16 IT: http://www.statmt.org/wmt16/it-translation-task.html

[3]WMT18: http://www.statmt.org/wmt18/translation-task.html

[4]https://github.com/alvations/sacremoses

B. Model Training and Hyperparameter Selection

To implement ReWE and ReSE, we have modified the popular OpenNMT open-source toolkit [44][5]. Two variants of the standard OpenNMT model have been used as baselines: the LSTM and the transformer, described hereafter.

LSTM: A strong NMT baseline was prepared by following the indications given by Denkowski and Neubig [42]. The model uses a bidirectional LSTM [45] for the encoder and a unidirectional LSTM for the decoder. The main hyperparameters such as the size of the word embeddings (300, 512, 796), the size of the hidden states (300, 512), the dropout rate (0.1, 0.2), the value of λ (0.1, 1, 2, 5, 10, 20), the value of β (2, 50, 100) and the learning rate (0.0002, 2) were tuned over the validation set, while the remaining hyperparameters were selected following the indications in [42]. All the selected hyperparameter values for each dataset are reported in Appendix A. As optimizer, we have used Adam [46]. During training, the learning rate was halved with simulated annealing upon convergence of the perplexity over the validation set, which was evaluated every $25,000$ training sentences. Training was stopped after halving the learning rate 5 times.

Transformer: The transformer network [4] has somehow become the *de-facto* neural network for the encoder and decoder of NMT pipelines thanks to its strong empirical accuracy and highly-parallelizable training. The same hyperparameters as for the LSTM were tuned on the validation set. The remaining hyperparameters were set to the default values of OpenNMT (more details are provided in Appendix A). With this model, we have not used simulated annealing since preliminary experiments showed that it did penalize performance. Training was stopped upon convergence in perplexity over the validation set, which was evaluated at every epoch for 20 epochs.

In addition, the word embeddings for both models were initialized with pre-trained fastText embeddings [47]. For the 300d word embeddings, we have used the word embeddings available on the official fastText website[6]. For the 512d embeddings and the subword units, we have trained our own pre-trained vectors using the fastText embedder with a large monolingual corpus from Wikipedia[7] and the training data. We have used pre-trained USE[8] and SBERT[9] sentence embedders with the datasets where English is the target language (i.e., De-En, Cs-En and Eu-En) because of their availability as monolingual encoders. When using BPE, the subwords of every sentence have been merged back into words before passing them to the USE. The performance of the BPE models has also been evaluated after post-processing the subwords back into words. Finally, hyperparameters λ and β have been tuned

[5]Our code is publicly available on GitHub at: https://github.com/ijauregiCMCRC/ReWE_and_ReSE.git. We will also release it on Code Ocean.

[6]fastText: https://fasttext.cc/docs/en/crawl-vectors.html

[7]Wikipedia: https://linguatools.org/tools/corpora/

[8]USE: https://tfhub.dev/google/universal-sentence-encoder/2

[9]SBERT: https://github.com/UKPLab/sentence-transformers

TABLE I: BLEU scores over the En-Fr test set. The reported results are the average of 3 independent runs and the standard deviation.

Models	Word	BPE
LSTM	34.29 ± 0.34	34.45 ± 0.06
+ ReWE	35.65† ± 0.35	35.47† ± 0.34
+ ReSE$_{avgEmbs}$	34.16 ± 0.42	34.36 ± 0.08
+ ReSE$_{maxpoolEmbs}$	34.41 ± 0.06	34.37 ± 0.14
+ ReWE + ReSE$_{avgEmbs}$	**35.80**† ± 0.30	**35.68**† ± 0.19
+ ReWE + ReSE$_{maxpoolEmbs}$	35.74† ± 0.29	35.52† ± 0.06
TRANS	33.73 ± 0.71	34.47 ± 0.52
+ ReWE	34.35 ± 0.55	34.89† ± 0.26
+ ReSE$_{avgEmbs}$	33.76 ± 0.59	34.12 ± 0.46
+ ReSE$_{maxpoolEmbs}$	33.37 ± 0.69	34.45 ± 0.62
+ ReWE + ReSE$_{avgEmbs}$	**34.56** ± 0.31	34.70 ± 0.43
+ ReWE + ReSE$_{maxpoolEmbs}$	34.36 ± 0.45	**34.98**† ± 0.26

TABLE II: BLEU scores over the Cs-En test set. The reported results are the average of 3 independent runs and the standard deviation.

Models	Word	BPE
LSTM	20.43 ± 0.24	22.36 ± 0.43
+ ReWE	21.90† ± 0.21	**23.75**† ± 0.26
+ ReSE$_{avgEmbs}$	20.31 ± 0.35	22.96† ± 0.18
+ ReSE$_{maxpoolEmbs}$	20.42 ± 0.53	22.48 ± 0.17
+ ReSE$_{USE}$	20.23 ± 0.24	22.71 ± 0.21
+ ReSE$_{SBERT}$	20.65† ± 0.14	22.62 ± 0.03
+ ReWE + ReSE$_{avgEmbs}$	21.60† ± 0.06	23.55† ± 0.59
+ ReWE + ReSE$_{maxpoolEmbs}$	21.47† ± 0.26	23.56† ± 0.11
+ ReWE + ReSE$_{USE}$	**22.11**† ± 0.33	23.40† ± 0.48
+ ReWE + ReSE$_{SBERT}$	21.47† ± 0.30	23.73† ± 0.36
TRANS	20.45 ± 0.71	20.80 ± 0.38
+ ReWE	21.08 ± 0.008	22.14† ± 0.35
+ ReSE$_{avgEmbs}$	19.81 ± 0.21	20.46 ± 0.26
+ ReSE$_{maxpoolEmbs}$	19.91 ± 0.25	20.85 ± 0.21
+ ReSE$_{USE}$	19.67 ± 0.19	20.88 ± 0.36
+ ReSE$_{SBERT}$	19.92 ± 0.13	20.91 ± 0.52
+ ReWE + ReSE$_{avgEmbs}$	21.07† ± 0.34	22.02† ± 0.45
+ ReWE + ReSE$_{maxpoolEmbs}$	20.80 ± 0.15	22.06† ± 0.23
+ ReWE + ReSE$_{USE}$	20.88 ± 0.26	21.79† ± 0.34
+ ReWE + ReSE$_{SBERT}$	**21.24**† ± 0.26	**22.20**† ± 0.18

TABLE III: BLEU scores over the Eu-En test set. The reported results are the average of 3 independent runs and the standard deviation.

Models	Word	BPE
LSTM	10.54 ± 0.17	17.11 ± 0.38
+ ReWE	13.33† ± 2.51	19.35† ± 0.94
+ ReSE$_{avgEmbs}$	13.60† ± 0.96	19.41† ± 0.71
+ ReSE$_{maxpoolEmbs}$	14.18† ± 2.58	17.94† ± 0.55
+ ReSE$_{USE}$	14.68† ± 0.36	17.50 ± 0.46
+ ReSE$_{SBERT}$	13.83† ± 2.06	18.38† ± 0.26
+ ReWE + ReSE$_{avgEmbs}$	16.44† ± 0.97	**21.24**† ± 0.14
+ ReWE + ReSE$_{maxpoolEmbs}$	**17.11**† ± 1.12	20.88† ± 0.57
+ ReWE + ReSE$_{USE}$	15.21† ± 0.68	20.29† ± 0.27
+ ReWE + ReSE$_{SBERT}$	15.41† ± 1.48	20.28† ± 0.99
TRANS	13.70 ± 1.11	13.42 ± 0.44
+ ReWE	**14.83**† ± 0.59	**14.55**† ± 0.77
+ ReSE$_{avgEmbs}$	11.71 ± 0.68	12.07 ± 1.31
+ ReSE$_{maxpoolEmbs}$	12.57 ± 0.77	10.98 ± 0.16
+ ReSE$_{USE}$	11.79 ± 0.38	10.42 ± 1.73
+ ReSE$_{SBERT}$	12.96 ± 1.32	10.86 ± 0.98
+ ReWE + ReSE$_{avgEmbs}$	14.03 ± 0.30	11.45 ± 0.69
+ ReWE + ReSE$_{maxpoolEmbs}$	14.20† ± 0.25	11.33 ± 1.09
+ ReWE + ReSE$_{USE}$	13.79 ± 0.98	11.65 ± 0.58
+ ReWE + ReSE$_{SBERT}$	13.69 ± 0.15	11.44 ± 0.10

only once for all datasets by using the En-Fr validation set. This was done in order to save the significant computational time that would have been required by further hyperparameter exploration. However, in the De-En case the initial results were far from the state of the art and we therefore repeated the selection with its own validation set. For all experiments, we have used an Intel Xeon E5-2680 v4 with an NVIDIA GPU card Quadro P5000. On this machine, the training time of the transformer has been approximately an order of magnitude larger than that of the LSTM.

C. Results

In this section, we report the results from a number of experiments carried out with both baselines and different combinations of ReWE and ReSE. The scores reported are the BLEU (BiLingual Evaluation Understudy) scores (in percentage points, or pp), a standard evaluation metric for machine translation [48]. The † symbol denotes statistically significant differences with respect to the baseline, computed with a corpus-level bootstrap significance test with p-value < 0.01 [49]. Table I shows the results over the En-Fr dataset. For this dataset, the proposed models have outperformed the LSTM and transformer baselines consistently by more than 1 BLEU pp. While applying ReSE alone has not improved the baseline models, jointly applying ReWE and ReSE has outperformed both the baselines and ReWE alone in all the cases. The sentence encoder that has achieved the highest score has been *avgEmbs* in all cases, except the transformer/BPE, where *maxpoolEmbs* has proved the best. Overall, the highest score reported by the proposed models (35.80, LSTM/Word) is $+1.33$ pp higher than the highest score achieved by the original models (34.47, Transformer/BPE) and sets the new state of the art for this dataset.

Table II reports the results over the Cs-En dataset. Also in this case, all the models with ReWE have improved over the corresponding baselines. For instance, LSTM/BPE+ReWE has achieved 23.75 BLEU pp, an improvement of $+1.39$ pp over the baseline. With this language pair, the transformer model has generally underperformed compared to the LSTM. In addition, this language pair has benefited more from the BPE pre-processing, most likely because Czech is a morphologically-rich language, with clearer subword divisions. Again, ReSE has generally improved performance when used in conjunction with ReWE compared to ReWE alone. Since the target language is English, on this dataset we have been able to use the USE and $SBERT$ sentence encoders which have led to mildly highers scores than the averaging/maxpooling of the word embeddings. Overall, the highest score reported by the proposed models has been 23.75 (LSTM/BPE), $+1.39$ pp higher than the highest score achieved by the original models

TABLE IV: BLEU scores over the De-En test set. The reported results are the average of 3 independent runs and the standard deviation.

Models	Word	BPE
LSTM	29.83 ± 0.05	**34.16** ± 0.10
+ ReWE	30.25† ± 0.35	33.89 ± 0.18
+ ReSE$_{avgEmbs}$	29.79 ± 0.20	33.49 ± 0.11
+ ReSE$_{maxpoolEmbs}$	29.68 ± 0.32	33.71 ± 0.40
+ ReSE$_{USE}$	29.70 ± 0.51	33.58 ± 0.38
+ ReSE$_{SBERT}$	29.84 ± 0.28	33.84 ± 0.38
+ ReWE + ReSE$_{avgEmbs}$	30.04 ± 0.21	33.80 ± 0.62
+ ReWE + ReSE$_{maxpoolEmbs}$	30.03 ± 0.30	33.73 ± 0.74
+ ReWE + ReSE$_{USE}$	30.16† ± 0.27	33.87 ± 0.63
+ ReWE + ReSE$_{SBERT}$	**30.31**† ± 0.24	33.31 ± 0.17
TRANS	29.68 ± 0.10	37.10 ± 0.21
+ ReWE	29.91† ± 0.20	36.98 ± 0.15
+ ReSE$_{avgEmbs}$	29.62 ± 0.48	36.88 ± 0.88
+ ReSE$_{maxpoolEmbs}$	29.28 ± 0.07	36.91 ± 0.09
+ ReSE$_{USE}$	29.27 ± 0.27	36.99 ± 0.21
+ ReSE$_{SBERT}$	29.41 ± 0.16	37.15 ± 0.23
+ ReWE + ReSE$_{avgEmbs}$	29.71 ± 0.27	37.12 ± 0.26
+ ReWE + ReSE$_{maxpoolEmbs}$	29.57 ± 0.31	37.03 ± 0.05
+ ReWE + ReSE$_{USE}$	29.53 ± 0.23	36.99 ± 0.05
+ ReWE + ReSE$_{SBERT}$	**30.41**† ± 1.39	**37.18** ± 0.20

TABLE V: BLEU scores over the En-Fr validation ("dev") set. The reported results are the average of 3 independent runs and the standard deviation.

Models	Word	BPE
LSTM	36.64 ± 0.26	37.09 ± 0.13
+ ReWE	38.03 ± 0.05	38.49 ± 0.15
+ ReSE$_{avgEmbs}$	36.77 ± 0.18	37.31 ± 0.27
+ ReSE$_{maxpoolEmbs}$	36.73 ± 0.32	37.16 ± 0.15
+ ReWE + ReSE$_{avgEmbs}$	**38.26** ± 0.26	**38.51** ± 0.23
+ ReWE + ReSE$_{maxpoolEmbs}$	38.07 ± 0.27	38.35 ± 0.23

TABLE VI: Training time per epoch over the Cs-En dataset.

Models	LSTM	Transformer
Baseline	727 s	403 s
+ ReWE	734 s	408 s
+ ReSE$_{avgEmbs}$	734 s	412 s
+ ReSE$_{maxpoolEmbs}$	736 s	413 s
+ ReSE$_{USE}$	1056 s	863 s
+ ReSE$_{SBERT}$	2555 s	3206 s
+ ReWE + ReSE$_{avgEmbs}$	1061 s	420 s
+ ReWE + ReSE$_{maxpoolEmbs}$	1059 s	419 s
+ ReWE + ReSE$_{USE}$	1037 s	844 s
+ ReWE + ReSE$_{SBERT}$	1053 s	3224 s

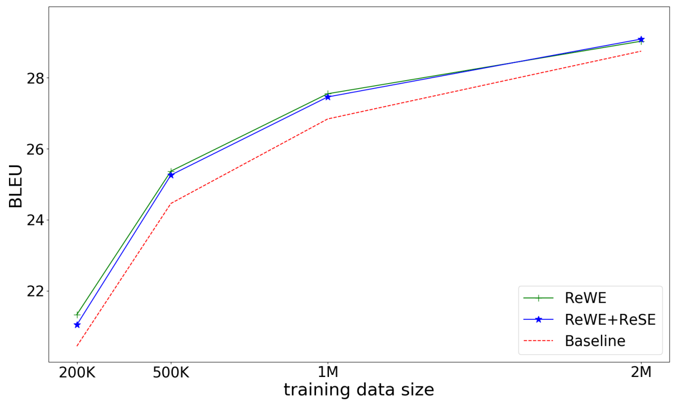


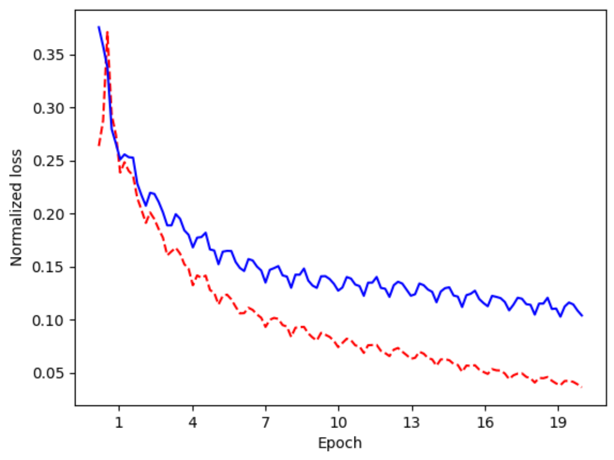
Fig. 3: BLEU scores over the De-En test set for models trained with training sets of different size.

and also the highest reported to date for this dataset.

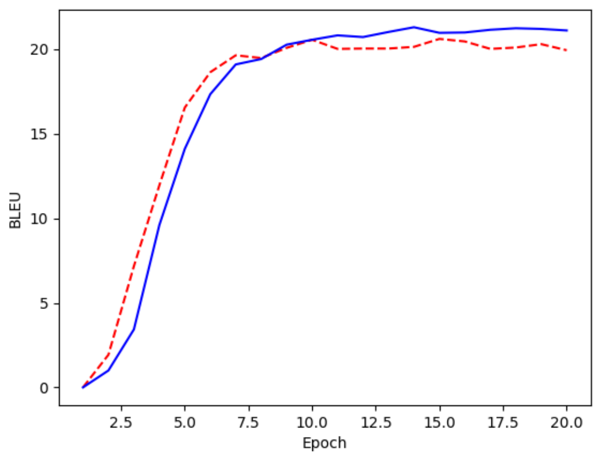
For the Eu-En dataset (Table III), the results show that, again, ReWE has outperformed both baselines by a large margin. Moreover, ReWE+ReSE has been able to improve the results even further (+4.13 BLEU pp when using BPE and +6.57 BLEU pp at word level over the corresponding baselines), again with the *avgEmbs* and *maxpoolEmbs* sentence encoders performing the best. Basque is, too, a morphologically-rich language and using BPE has proved very beneficial (+4.27 BLEU pp over the best word-level model). As noted before, the Eu-En dataset is very low-resource ($<$ 90K sentence pairs) which makes it more likely for the baselines to generalize poorly. Consequently, regularizers such as ReWE and ReSE are more helpful, with larger margins of improvement compared to other datasets. On a separate note, the transformer has unexpectedly performed well below the LSTM on this dataset, and especially so with BPE. We speculate that the transformer may have suffered more from the small size of this training set. Overall, the highest score reported by the proposed models (LSTM/BPE) has been +4.13 higher than the highest score achieved by the baselines, setting a new state-of-the-art accuracy also for this dataset.

Finally, Table IV shows the results over the De-En dataset that we categorize as high-resource (5M+ sentence pairs). On this dataset both ReWE and ReWE+ReSE have been able to improve the results of the baselines, although the margins of improvement have been smaller than for the other language pairs. In the case of the transformer/BPE, the baseline itself has achieved a very high score of 37.10 BLEU pp, and the best proposed regularization combination (ReWE + ReSE SBERT) has only improved it by 0.08 pp. This shows that when the training data are abundant, the proposed regularizers may not be beneficial. To probe this further, we have repeated these experiments by training the models over subsets of the training set of increasing size (200K, 500K, 1M, and 2M sentence pairs). Fig. 3 shows the BLEU scores achieved by the baseline and the regularized models for the different training data sizes. The plot clearly shows that the performance margin increases as the training data size decreases, as expected from a regularized model.

On the whole, we have observed no single "best" among the different sentence encoders, as their relative performance has varied across the different datasets. We hypothesize that the sentence embedders have better performance in more general translation domains such as TED talks (Cs-En dataset) and news (De-En dataset) than in specialized domains such as IT (Eu-En dataset). In any case, the choice of sentence encoder needs to be addressed as a discrete hyperparameter, selecting the best encoder based on the performance on the validation set. As an example, Table V shows the results for the LSTM model over the En-Fr validation set, where it can be seen that the best model is the same as the best model over the test



(a) Normalized training loss at different training steps.



(b) BLEU score over the test set, at different training steps.

Fig. 4: Impact of the proposed regularizers: the plots compare a transformer baseline (red dashed line) with a regularized model using ReWE + ReSE$_{SBERT}$ (blue continuous line) on the Cs-En dataset; a) training loss at successive epochs; b) test-set BLEU scores at the corresponding epochs. The plots show that the regularized model is able to mollify overfitting and surpass the baseline in test-set performance.

set (Table I). The relative rankings of the other models are also very similar, showing that the selection of the sentence encoder using the validation set is a viable approach.

A possible further factor to take into account for model selection is the model's training time. As an example, Table VI shows the time per epoch for each model over the Cs-En training set. When using ReWE and ReSE with simple sentence encoders (*avgEmbs* and *maxpoolEmbs*), the training time has barely increased compared to the baselines. However, when using them with the more complex sentence encoders (*USE* and *SBERT*) the training times have increased considerably, especially with the transformer.

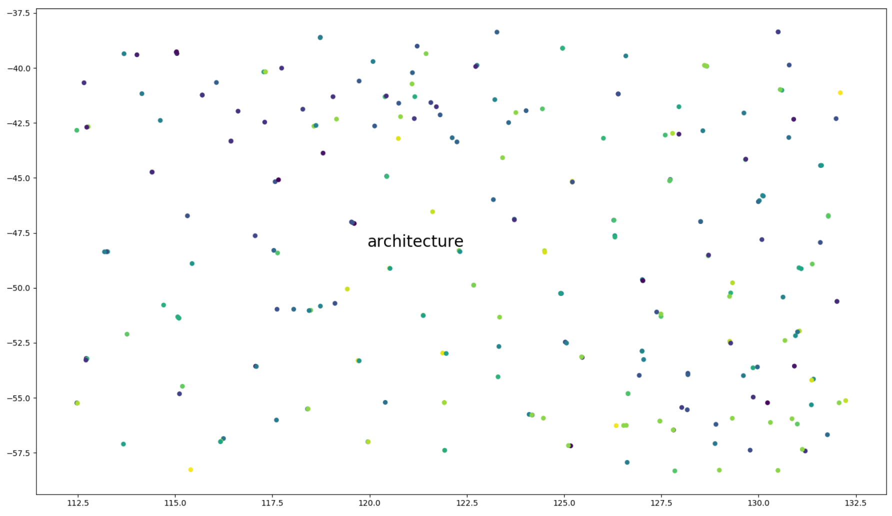
To further illustrate the impact of the proposed regularizers, Figure 4 compares a transformer baseline and a regularized model as the training progresses. Figure 4a plots the value of the training loss at successive training epochs. The plots show that the training loss of the regularized model decreases more slowly, possibly indicating that the regularized model is able to avoid overfitting the training set. In turn, Figure 4b plots the test-set BLEU scores of the partially-trained models at corresponding training epochs. The plots show that the regularized model initially achieves lower BLEU scores, but at a certain point it manages to surpass the baseline, confirming its ability to escape overfitting.

For a qualitative evaluation of the translations, Table VII shows two examples for the Eu-En and Cs-En language pairs. The examples show that both ReWE alone and ReWE+ReSE have improved the quality of these translations. For instance, in the Eu-En example ReWE has correctly translated "File tab", and ReWE+ReSE has correctly added "click Create". In the Cs-En example, the ReWE model has picked the correct subject, "they", yet only the ReWE+ReSE model has correctly translated "students" and captured the opening phrase "What was…about this…".

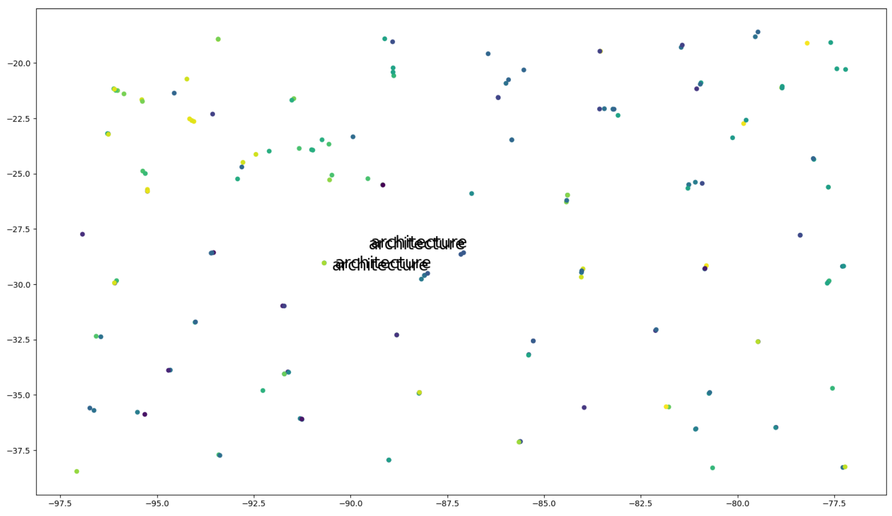
D. Analyzing the Behavior of the Proposed Regularizers

The quantitative experiments have proven that ReWE and ReSE can act as effective regularizers for low-resource NMT. Yet, it would be useful to understand how they influence the training to achieve regularization. To this aim, we have explored the values of the hidden vectors on the decoder end (vector $\mathbf{s}_j$, Eq. 5). These values are the "feature space" used by the final classification block (a linear transformation and a softmax) and can provide insights into the model. For this reason, we have stored all the $\mathbf{s}_j$ vectors with their respective word predictions for the LSTM models on the Cs-En test set. Then, we have used t-SNE [50] to reduce the dimensionality of the $\mathbf{s}_j$ vectors to a visualizable two dimensions. Finally, we have chosen a particular word (*architecture*) as the center of the visualization, and plotted all the vectors within a chosen neighborhood of this center word (Fig. 5). To avoid cluttering the figure, we have not superimposed the predicted words to the vectors, but only used a different color for each distinct word (this figure should be viewed in color). The center word in the two subfigures (a: baseline; b: baseline + ReWE) is the same and from the same source sentence, so the visualized regions can be compared. The visualizations also display all other predicted instances of word *architecture* within the neighborhood.

These visualizations show two interesting behaviors: 1) from eye judgment, the points predicted by the ReWE model seem more uniformly spread out; 2) instances of the same words have $\mathbf{s}_j$ vectors that are close to each other. For instance, several instances of word *architecture* are close to each other in Fig. 5b while no other instance other than the reference appears in Fig. 5b. The overall observation is that the ReWE regularizer leads to a vector space that is easier to discriminate (i.e., find class boundaries for), facilitating the final word prediction. In order to confirm this observation, we have also



(a) Baseline



(b) Baseline + ReWE

Fig. 5: Visualization of the $\mathbf{s}_j$ vectors from the decoder for a subset of the Cs-En test set. Please refer to Section V-D for explanations. This figure should be viewed in color.

computed various clustering indexes over the clusters formed by the vectors with identical predicted word. As indexes, we have used the *silhouette* and Davies-Bouldin indexes that are two well-known unsupervised metrics for clustering. The silhouette index ranges from -1 to +1, where values closer to 1 mean that the clusters are compact and well separated. The Davies-Bouldin index is an unbounded nonnegative value, with values closer to 0 meaning better clustering. Table VIII shows the values of these clustering indexes over the Cs-En test set. As the table shows, the models with ReWE and ReWE+ReSE have reported the best values. This confirms that applying these regularizers has a positive impact on the decoder's hidden space, ultimately justifying the increase in translation accuracy.

For further exploration, we have created another visualization of the $\mathbf{s}_j$ vectors and their predictions over a smaller neighborhood (Fig. 6). The same word (*architecture*) has been used as the center word of the plot. Then, we have "vibrated" each of the $\mathbf{s}_j$ vector by small random increments (uniformly in interval [0.05, 8]) in each of their dimensions, creating several new synthetic instances of $\mathbf{s}$ vectors which are very close to the original ones. All these synthetic vectors have

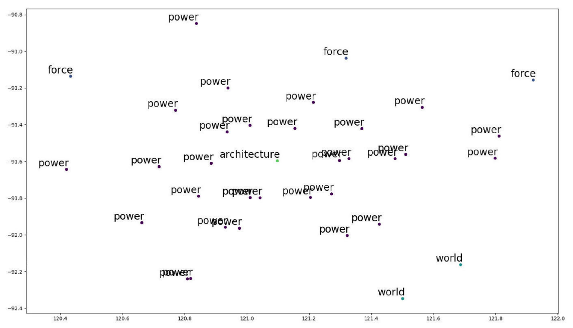
TABLE VII: Translation examples. Example 1: Eu-En; Example 2: Cs-En. For these examples, we have used the best performing sentence embedder over the validation set (i.e., maxpoolEmbs for Eu-En and SBERT for Cs-En).

Example 1:	
Src:	Sakatu Fitxategia fitxa Oihal atzeko ikuspegia atzitzeko ; sakatu Berria . Hautatu txantiloia eta sakatu Sortu hautatutako txantiloia erabiltzeko .
Ref:	Click the File tab to access Backstage view , select New . Select a template and click Create to use the selected template .
Baseline:	Click the default tab of the tab that you want to open the tab tab . Select the template and select the selected template .
Baseline + ReWE:	Press the File tab to access the view view ; click New . Select the template and click Add to create the selected template .
Baseline + ReWE + ReSE:	Press the File tab to access the chart view ; press New . Select the template and click Create to use the selected template .
Example 2:	
Src:	Na tomto projektu bylo skvělé , že žáci viděli lokální problém a bum – okamžitě se s ním snaží vyrovnat .
Ref:	What was really cool about this project was that the students saw a local problem , and boom – they are trying to immediately address it .
Baseline:	In this project , it was great that the kids had seen local problems and boom – immediately he's trying to deal with him .
Baseline + ReWE:	In this project , it was great that the kids saw a local issue , and boom – they immediately try to deal with it .
Baseline + ReWE + ReSE:	What was great about this project was that the students saw a local problem, and boom , they're trying to deal with him .

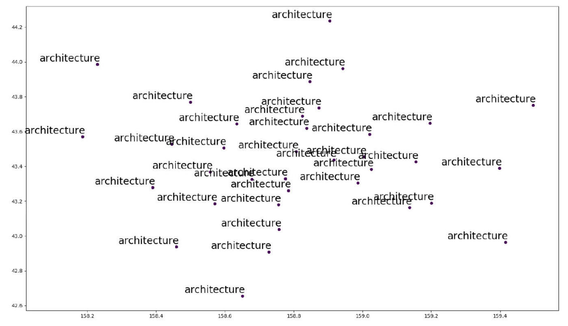
TABLE VIII: Clustering indexes of the LSTM models over the Cs-En test set. The reported results are the average of 5 independent runs.

Model	Sillhouette	Davies-Bouldin
LSTM	-0.19	1.87
+ ReWE ($\lambda = 2$)	-0.17	**1.80**
+ ReWE ($\lambda = 2$) + ReSE ($\beta = 2$)	**-0.16**	**1.80**

then been classified with the trained NMT models to obtain the corresponding word predictions. Finally, we have used t-SNE to reduce the dimensionality to 2d, and visualized all the vectors and their predictions in a small neighborhood (± 10 units) around the center word. Fig. 6 shows that, with the ReWE model, all the vectors surrounding the center word have consistently predicted the same word (*architecture*). Conversely, with the baseline, the surrounding points have predicted different words (*power*, *force*, *world*). This is additional evidence that the hidden $\mathbf{s}$ space is evened out by the use of the proposed regularizers.



(a) Baseline



(b) Baseline + ReWE

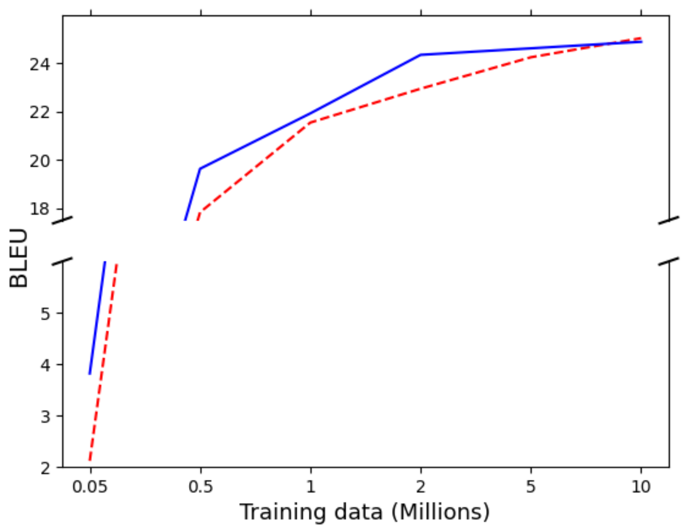
Fig. 6: Visualization of the $\mathbf{s}_j$ vectors in a smaller neighborhood of the center word.

E. Experiments with Unsupervised NMT

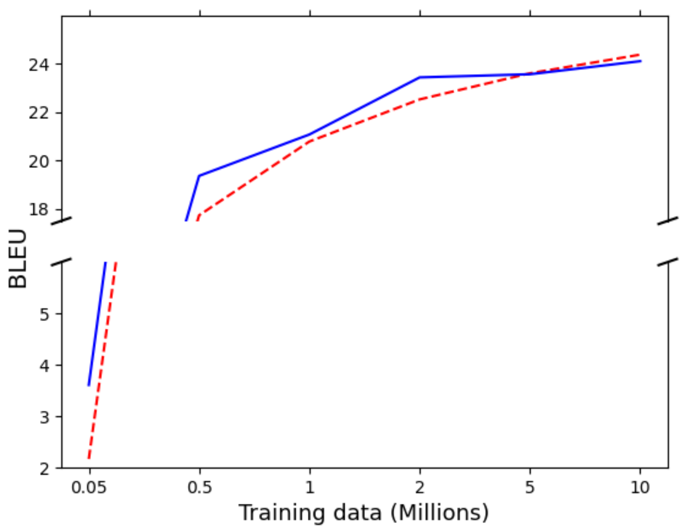
As a last experiment, we have employed the proposed regularizers for an unsupervised NMT task. For this experiment, we have used the open-source model provided by Lample *et al.* [34][10] which is currently the state of the art for unsupervised NMT, and also adopted its default hyperparameters and preprocessing steps which include 4-layer transformers for the encoder and both decoders (source and target languages), and BPE subword learning. The experiments have been performed by training the models with the monolingual data of the WMT14 English-French training set, and by testing them on the test set in both language directions (En-Fr and Fr-En).

As described in Section II-C, an unsupervised NMT model includes two decoders to be able to translate into both languages. The model is trained by iterating over two alternate steps: 1) training using the decoders as monolingual, denoising language models (e.g., En-En, Fr-Fr), and 2) training using back-translations (e.g., En-Fr-En, Fr-En-Fr). This means that an unsupervised NMT model uses a total of four different objective functions across the two steps and the two translation directions. Potentially, various regularizer combinations could

[10]UnsupervisedMT: https://github.com/facebookresearch/UnsupervisedMT



(a) En-Fr



(b) Fr-En

Fig. 7: BLEU scores over the test set. The reported results are the average of 5 independent runs. The red dashed line represents the baseline model and the blue continuous line represents the baseline + ReWE model.

be used with each of them. However, for simplicity we have decided to limit this experiment to only the ReWE regularizer and the back-translation step, in both directions. We plan to experiment with the use of ReSE in future work. The balancing hyperparameter, λ, was tuned over the validation set and eventually set to 0.2.

Fig. 7 shows the results from the different models trained with increasing amounts of monolingual data (50K, 500K, 1M, 2M, 5M and 10M sentences in each language). The model trained using ReWE has been able to consistently outperform the baseline in both language directions. The trend we had observed in the supervised case has also applied to these experiments: the performance margin has been larger for smaller training data sizes, and has reduced as the training size increased. For example, in the En-Fr direction the margin has only been $+0.44$ BLEU points when training with 10M sentences, but has been $+1.74$ BLEU points when training with 50K sentences. Again, these results are in line with the behavior of an effective regularized objective.

VI. Conclusion

In this paper, we have presented a regularization approach for improving the performance of NMT models in low-resource scenarios. The approach is based on regressing continuous representations of words and sentences (nicknamed ReWE and ReSE, respectively) during training to improve the models' generalization beyond their parallel training data. Extensive experiments over three low-resource datasets ($89-220$K parallel sentences) and a variable-size dataset (200K-2M parallel sentences) have shown that both ReWE and ReWE+ReSE have significantly improved the performance of the NMT models, for increases in BLEU score of up to 4.13 percentage points over the highest scores obtained by the non-regularized models, and of up 6.57 percentage points over a corresponding baseline. To analyze the behavior of the proposed regularizers, we have also presented a detailed analysis showing how the regularization impacts the decoder's output space, enhancing the clustering of the vectors associated with unique words. Finally, we have shown that the regularized models have also outperformed the baselines in an experiment on unsupervised NMT, where no parallel data are required. As future work, we plan to explore whether the categorical and continuous predictions from our model could be jointly utilized also at inference time to further refine the quality of the translations.

References

[1] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, pp. 3104–3112, 2014.

[2] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *Proc. Int. Conf. Learn. Representations (ICLR)*, 2015.

[3] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, pp. 1412–1421, 2015.

[4] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, pp. 5998–6008, 2017.

[5] P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, *et al.*, "Moses: Open source toolkit for statistical machine translation," in *Proc. 45th Annu. Meeting Assoc. Comput. Linguistics (ACL)*, pp. 177–180, 2007.

[6] J. Hu, S. Ruder, A. Siddhant, G. Neubig, O. Firat, and M. Johnson, "XTREME: A massively multilingual multi-task benchmark for evaluating cross-lingual generalisation," in *Proc. 37th Int. Conf. Mach. Learn. (ICML)*, vol. 119, pp. 4411–4421, 2020.

[7] M. Elbayad, L. Besacier, and J. Verbeek, "Token-level and sequence-level loss smoothing for RNN language models," in *Proc. 56th Annu. Meeting Assoc. Comput. Linguistics (ACL)*, pp. 2094–2103, 2018.

[8] P. Koehn and R. Knowles, "Six challenges for neural machine translation," in *Proc. First Workshop Neural Machine Translation*, pp. 28–39, Assoc. Comput. Linguistics (ACL), 2017.

[9] I. Jauregi Unanue, E. Zare Borzeshi, N. Esmaili, and M. Piccardi, "ReWE: Regressing word embeddings for regularization of neural machine translation systems," in *Proc. North American Chapter Assoc. Comput. Linguistics (NAACL)*, 2019.

[10] D. Cer, Y. Yang, S.-y. Kong, N. Hua, N. Limtiaco, R. S. John, N. Constant, M. Guajardo-Cespedes, S. Yuan, C. Tar, *et al.*, "Universal sentence encoder," *arXiv:1803.11175 [cs]*, 2018.

[11] N. Reimers and I. Gurevych, "Sentence-bert: Sentence embeddings using siamese bert-networks," *arXiv preprint arXiv:1908.10084*, 2019.

[12] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[13] Y. Gal and Z. Ghahramani, "A theoretically grounded application of dropout in recurrent neural networks," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, pp. 1019–1027, 2016.

[14] M. Fadaee, A. Bisazza, and C. Monz, "Data augmentation for low-resource neural machine translation," in *Proc. 55th Annu. Meeting Assoc. Comput. Linguistics (ACL)*, pp. 567–573, 2017.

[15] J. Gu, Y. Wang, Y. Chen, K. Cho, and V. O. Li, "Meta-learning for low-resource neural machine translation," *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2018.

[16] K. Clark, M.-T. Luong, C. D. Manning, and Q. V. Le, "Semi-supervised sequence modeling with cross-view training," *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2018.

[17] R. Sennrich, B. Haddow, and A. Birch, "Improving neural machine translation models with monolingual data," *Proc. 54th Annu. Meeting Assoc. Comput. Linguistics (ACL)*, 2016.

[18] S. Edunov, M. Ott, M. Auli, and D. Grangier, "Understanding back-translation at scale," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, pp. 489–500, 2018.

[19] T. Kudo, "Subword regularization: Improving neural network translation models with multiple subword candidates," in *Proc. 56th Annu. Meeting Assoc. Comput. Linguistics (ACL)*, pp. 66–75, 2018.

[20] K. Chousa, K. Sudoh, and S. Nakamura, "Training neural machine translation using word embedding-based loss," *arXiv:1807.11219 [cs]*, 2018.

[21] S. Ma, X. Sun, Y. Wang, and J. Lin, "Bag-of-words as target for neural machine translation," in *Proc. 56th Annu. Meeting Assoc. Comput. Linguistics (ACL)*, pp. 332–338, 2018.

[22] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *Proc. 26th Int. Conf. Mach. Learn. (ICML)*, pp. 41–48, 2009.

[23] T. Kocmi and O. Bojar, "Curriculum learning and minibatch bucketing in neural machine translation," in *Proc. Int. Conf. Recent Adv. Natural Lang. Process. (RANLP)*, pp. 379–386, 2017.

[24] E. A. Platanios, O. Stretcu, G. Neubig, B. Poczos, and T. Mitchell, "Competence-based curriculum learning for neural machine translation," in *Proc. North American Chapter Assoc. Comput. Linguistics (NAACL)*, pp. 1162–1172, 2019.

[25] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," *Proc. North American Chapter Assoc. Comput. Linguistics (NAACL)*, 2018.

[26] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," *Proc. North American Chapter Assoc. Comput. Linguistics (NAACL)*, 2019.

[27] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding by generative pre-training," *URL https://s3-us-west-2. amazonaws. com/openai-assets/research-covers/languageunsupervised/language understanding paper. pdf*, 2018.

[28] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *Journal of Machine Learning Research*, 2019.

[29] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, pp. 3111–3119, 2013.

[30] J. Pennington, R. Socher, and C. Manning, "GloVe: Global vectors for word representation," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, pp. 1532–1543, 2014.

[31] M. Artetxe and H. Schwenk, "Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond," *arXiv:1812.10464 [cs]*, 2018.

[32] S. Edunov, A. Baevski, and M. Auli, "Pre-trained language model representations for language generation," *Proc. North American Chapter Assoc. Comput. Linguistics (NAACL)*, 2019.

[33] S. Kumar and Y. Tsvetkov, "Von Mises-Fisher loss for training sequence to sequence models with continuous outputs," *Proc. Int. Conf. Learn. Representations (ICLR)*, 2018.

[34] G. Lample, M. Ott, A. Conneau, L. Denoyer, and M. Ranzato, "Phrase-based & neural unsupervised machine translation," *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2018.

[35] M. Artetxe, G. Labaka, E. Agirre, and K. Cho, "Unsupervised neural machine translation," *Proc. Int. Conf. Learn. Representations (ICLR)*, 2017.

[36] Z. Yang, W. Chen, F. Wang, and B. Xu, "Unsupervised neural machine translation with weight sharing," *Proc. 56th Annu. Meeting Assoc. Comput. Linguistics (ACL)*, 2018.

[37] A. Sharaf, H. Hassan, and H. Daumé III, "Meta-learning for few-shot NMT adaptation," in *Proc. Fourth Workshop on Neural Generation and Translation*, pp. 43–53, 2020.

[38] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proc. 25th Int. Conf. Mach. Learn. (ICML)*, pp. 1096–1103, 2008.

[39] B. Ondrej, R. Chatterjee, F. Christian, G. Yvette, H. Barry, H. Matthias, K. Philipp, L. Qun, L. Varvara, M. Christof, *et al.*, "Findings of the 2017 conference on machine translation (WMT17)," in *Proceedings of the Second Conference on Machine Translation*, pp. 169–214, 2017.

[40] O. Bojar, C. Federmann, M. Fishel, Y. Graham, B. Haddow, P. Koehn, and C. Monz, "Findings of the 2018 conference on machine translation (WMT18)," in *Proceedings of the Third Conference on Machine Translation*, pp. 272–303, 2018.

[41] L. Barrault, O. Bojar, M. R. Costa-Jussà, C. Federmann, M. Fishel, Y. Graham, B. Haddow, M. Huck, P. Koehn, S. Malmasi, *et al.*, "Findings of the 2019 conference on machine translation (WMT19)," in *Proceedings of the Fourth Conference on Machine Translation*, pp. 1–61, 2019.

[42] M. Denkowski and G. Neubig, "Stronger baselines for trustable results in neural machine translation," in *Proc. First Workshop Neural Machine Translation*, pp. 18–27, Assoc. Comput. Linguistics (ACL), 2017.

[43] R. Sennrich, B. Haddow, and A. Birch, "Neural machine translation of rare words with subword units," in *Proc. 54th Annu. Meeting Assoc. Comput. Linguistics (ACL)*, pp. 1715–1725, 2016.

[44] G. Klein, Y. Kim, Y. Deng, J. Senellart, and A. M. Rush, "OpenNMT: Open-source toolkit for neural machine translation," *arXiv:1701.02810 [cs]*, 2017.

[45] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[46] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Representations (ICLR)*, 2015.

[47] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *Trans. Assoc. Comput. Linguistics (TACL)*, vol. 5, pp. 135–146, 2017.

[48] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "BLEU: a method for automatic evaluation of machine translation," in *Proc. 40th Annu. Meeting Assoc. Comput. Linguistics (ACL)*, 2002.

[49] R. Dror, G. Baumer, S. Shlomov, and R. Reichart, "The hitchhiker's guide to testing statistical significance in natural language processing," in *Proc. 56th Annu. Meeting Assoc. Comput. Linguistics (ACL)*, pp. 1383–1392, 2018.

[50] L. v. d. Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, no. Nov, pp. 2579–2605, 2008.

Inigo Jauregi Unanue received the BEng degree in telecommunication systems from University of Navarra, Donostia-San Sebastian, Spain, in 2016, and the PhD degree in natural language processing from University of Technology Sydney in 2020. From 2014 to 2016, he was a research assistant at Centro de Estudio e Investigaciones Tecnicas (CEIT). Currently, he is a natural language processing and machine learning researcher at RoZetta Technology in Sydney, Australia. His research interests include machine translation and low-resource natural language processing.

Ehsan Zare Borzeshi received the PhD degree from University of Technology Sydney, Australia, in 2013. He is currently a Senior Data & Applied Scientist with Microsoft CSE (Commercial Software Engineering). He has previously held appointments as a senior researcher at the University of Newcastle, University of Technology Sydney, and the RoZetta Institute (formerly CMCRC) in Sydney. He has also been a Visiting Scholar with the University of Central Florida, Orlando, FL, USA. His current research interests include big data, deep learning and natural language processing.

Massimo Piccardi (SM'05) received the MEng and PhD degrees from the University of Bologna, Bologna, Italy, in 1991 and 1995, respectively. He is currently a Full Professor of computer systems with University of Technology Sydney, Australia. His research interests include natural language processing, computer vision and pattern recognition and he has co-authored over 180 papers in these areas. Prof. Piccardi is a Senior Member of the IEEE, a member of its Computer and Systems, Man, and Cybernetics Societies, and a member of the International Association for Pattern Recognition. He presently serves as an Associate Editor for the IEEE Transactions on Big Data.

Appendix A
Hyperparameters

As mentioned in Section V-B, hyperparameter selection as been performed over the validation set for each model. Tables IX-XII show the final selected values for each dataset and model type (i.e., LSTM or transformer) in order to facilitate the reproducibility of all the results reported in the paper.

TABLE IX: Hyperparameters for the En-Fr dataset.

Hyperparameter	LSTM	Transformer
encoder	Bi-LSTM	Transformer
decoder	LSTM	Transformer
word embedding dim	300	512
hidden layer dim	1024	512
# layers	2	6
global attention	*mlp*	*general*
head count	—	8
position encoding	—	True
dropout	0.1	0.1
label smoothing	0	0.1
batch type	*sentences*	*tokens*
batch size	32	4096
normalization	*sentences*	*tokens*
gradient accumulation	1	4
optimizer	*Adam*	*Adam*
decay method	—	*noam*
learning rate	0.0002	2
glorot initialization	—	True
warmup steps	0	8000
ReWE λ	20	20
ReSE β	100	100

TABLE X: Hyperparameters for the Cs-En dataset.

Hyperparameter	LSTM	Transformer
encoder	Bi-LSTM	Transformer
decoder	LSTM	Transformer
word embedding dim	300	300
hidden layer dim	1024	300
# layers	2	6
global attention	*mlp*	*general*
head count	—	6
position encoding	—	True
dropout	0.1	0.1
label smoothing	0	0.1
batch type	*sentences*	*tokens*
batch size	32	1024
normalization	*sentences*	*tokens*
gradient accumulation	1	4
optimizer	*Adam*	*Adam*
decay method	—	*noam*
learning rate	0.0002	2
glorot initialization	—	True
warmup steps	0	8000
ReWE λ	20	20
ReSE β	100	100

TABLE XI: Hyperparameters for the Eu-En dataset.

Hyperparameter	LSTM	Transformer
encoder	Bi-LSTM	Transformer
decoder	LSTM	Transformer
word embedding dim	300	300
hidden layer dim	1024	300
# layers	2	6
global attention	*mlp*	*general*
head count	—	6
position encoding	—	True
dropout	0.1	0.1
label smoothing	0	0.1
batch type	*sentences*	*tokens*
batch size	32	256
normalization	*sentences*	*tokens*
gradient accumulation	1	4
optimizer	*Adam*	*Adam*
decay method	—	*noam*
learning rate	0.0002	2
glorot initialization	—	True
warmup steps	0	8000
ReWE λ	20	20
ReSE β	100	100

TABLE XII: Hyperparameters for the De-En dataset.

Hyperparameter	LSTM	Transformer
encoder	Bi-LSTM	Transformer
decoder	LSTM	Transformer
word embedding dim	300	300
hidden layer dim	1024	300
# layers	2	6
global attention	*mlp*	*general*
head count	—	6
position encoding	—	True
dropout	0.1	0.1
label smoothing	0	0.1
batch type	*sentences*	*tokens*
batch size	32	1024
normalization	*sentences*	*tokens*
gradient accumulation	1	4
optimizer	*Adam*	*Adam*
decay method	—	*noam*
learning rate	0.0002	2
glorot initialization	—	True
warmup steps	0	8000
ReWE λ	2	2
ReSE β	2	2