RIGA at SemEval-2022 Task 1: Scaling Recurrent Neural Networks for CODWOE Dictionary Modeling

Eduards Mukans
mukans.work@gmail.com
University of Latvia

Gus Strazds
gs15014@students.lu.lv
University of Latvia

Guntis Barzdins
guntis@latnet.lv
University of Latvia, IMCS

Abstract

Described are our two entries "emukans" and "guntis" for the definition modeling track of CODWOE SemEval-2022 Task 1. Our approach is based on careful scaling of a GRU recurrent neural network, which exhibits double descent of errors, corresponding to significant improvements also per human judgement. Our results are in the middle of the ranking table per official automatic metrics.

1 Introduction

The definition modeling track of SemEval-2022 Task 1: CODWOE - COmparing Dictionaries and WOrd Embeddings (Mickus et al., 2022) challenged participants to generate dictionary glosses from individual word embedding vectors. This paper describes two CODWOE submissions, "emukans" and "guntis", where the first focuses on the automatic CODWOE scores, but the second attempts to gauge the relationships between scaling laws, the automated metrics, and human evaluation. Our submissions achieved competitive results (see Figure 3) on the MoverScore official metric - scoring 1st for French, 2nd for Spanish, and 3rd for Russian.

Our approach was to apply classical recurrent networks, such as Long Short-term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) and Gated Recurrent Units (GRU) (Cho et al., 2014), to definition modeling and investigate how model scaling impacts performance. The scaling effect is well investigated for transformers, but not so much for RNNs. Recently, the main focus in deep learning has skewed from searching for new model architectures to investigating how various factors impact the training process and overall system performance (Nakkiran et al., 2019; Kaplan et al., 2020; Gordon et al., 2021). The main factors are: the amount of data, the amount of compute, and the size of the model (parameter count). In the competition the data amount is fixed and no use of external data is permitted, thus we investigate how scaling model size and training time impacts training progress and model performance for recurrent models.

In our experiments we did observe deep double descent effects: epoch-wise double descent with respect to both cross-entropy loss and prediction accuracy on a validation data set, more pronounced with increasing model size.

We also investigated the automatic metrics used for evaluating submissions and their correlation with human evaluation, focusing primarily on the MoverScore metric (Wei Zhao, 2019). MoverScore does correlate with human evaluation, but not necessarily very strong, at least for this dataset. We find that the double descent effect seen with respect to prediction accuracy can also be observed for MoverScore.

2 Background

The CODWOE shared task invites participants to compare two types of semantic descriptions: dictionary glosses and word embedding representations. The task consists of 2 subtracks: definition modeling and reverse dictionary. In definition modeling participants have to generate glosses from word embedding representations. The reverse dictionary task is the inverse: reconstruct a word embedding from the corresponding gloss. Considering results achieved by the baseline models provided by the organisers, we decided to participate only in the definition modeling track, as it seems the more challenging task, with more room for potential improvement.

For the definition modeling track, inputs are 256-dimensional embedding vectors and outputs are plain text. Data is provided for 5 languages: English, French, Spanish, Italian and Russian. Every language is scored separately. We submitted for all 5 languages. The provided word embedding vec-

Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022), pages 82 - 87

tors are of 3 types: CHAR, SGNS, and ELECTRA.

3 System overview

For the definition modeling task, we used classical recurrent networks, experimenting with both LSTM and GRU architectures. We added an initial fully connected input layer to scale a given word embedding vector to higher dimensions. We used the ADAM optimizer (Kingma and Ba, 2014) in the training process. The learning rate was set in the range $\in$ [1e-5, 3e-5, 1e-4]. A linear learning rate decay schedule with warm-up over 0.01 was used. No preprocessing was applied to training data. The code is available on GitHub [1]

The very first step is creating a tokenizer and building its vocabulary. We use SentencePiece tokenization (Kudo and Richardson, 2018), trained on the training dataset only. We carried out experiments across a range of vocabulary sizes.

We used a classical approach and a decoder only part of standard seq2seq (sequence to sequence) recurrent neural network models without attention. The GRU/LSTM state vector is initialized from the given defmod embedding vector. In our case, we use a single word embedding vector type. For the first time step we pass a single <seq> token to model as input. The model outputs a single predicted token and a new state vector. To avoid exploding gradients, the outputs of the model are normalized. The token selected by the model is appended to the generated gloss, and is also used as input to the model for the next time step. This process is repeated until reaching the iteration limit.

At every time step, the model can make mistakes. If the initial part of the input sequence fed to a seq2seq model is bad, most likely the subsequent output sequence will also be wrong. To mitigate this accumulation of errors and speed up the training process, we use the teacher forcing technique (Williams and Zipser, 1989). With teacher forcing, the model is trained by supplying input tokens from the target sequence of the dataset and using the network's one-step-ahead predictions to do multi-step sampling. We also tried a more advanced teacher forcing technique: scheduled sampling (Duckworth et al., 2020), where input sequence tokens are given ground-truth values only with some probability. Unfortunately, scheduled sampling did not give good results - the loss plot was very noisy. It is likely that the CODWOE definition modeling task itself is a very hard task with too much variability relative to the amount of provided training data; scheduled sampling might be better suited for language model fine-tuning when the model weights are pretrained on a large corpus and already correlate fairly well with natural language syntax and semantics.

After each training epoch, the model is evaluated on a validation dataset using the same cross-entropy loss function as used for training. We also use an accuracy metric for evaluating model performance, as it correlates with perplexity and human judgement for large language models. The accuracy is calculated by dividing the count of correctly predicted tokens (under teacher-forcing) by the number of total tokens.

For "emukans" submissions, model training is stopped using early stopping (Prechelt, 2012) based on the accuracy score for the validation data, while "guntis" submissions were intentionally trained long past the overfitting point to observe scaling and double descent effects.

4 Experimental setup

For our experiments, we have 5 Tesla v100 16GB GPUs provided by our institute. During the competition, our focus was on exploring different training effects and model tuning. Most of the experiments were focused on primary factors of "scaling laws": model size and the amount of compute (training epochs).

For simplicity and consistency of presentation, in most of the following tables and figures (all except for Figure 3) we report experimental performance evaluated against a trial dataset provided by the CODWOE organizers, which consists of only 200 glosses. Apart from the automatic metrics, our focus was on (informal) manual evaluation of generated glosses.

4.1 Vocabulary size

The vocabulary of distinct tokens available for use by an NLP model is generally built during a data preparation stage, and the size of this vocabulary is a key factor in model performance. Therefore we started our experiments by tuning the size of the vocabulary.

We build our token vocabulary from the training dataset only. Taking into account the relatively small training dataset - only 43k glosses and 18k unique words, we reasoned that the token vocabu-

[1] https://github.com/emukans/codwoe

Vocab size	MoverScore	S-BLEU	L-BLEU
250	0.09702	0.02504	0.02508
500	0.10662	0.02452	0.02455
800	0.11754	0.02469	0.02470
1500	0.13045	**0.02726**	**0.02726**
3000	0.13379	0.02679	0.02681
5000	**0.13625**	0.02593	0.02596
15000	0.09638	0.02053	0.02056

Table 1: Influence of vocabulary size on the automatic metrics

lary size should be fairly small. Therefore we set our hypothesis as the following:

Hypothesis 1 (H1): *Optimal vocabulary size is around 10% of the unique word tokens.*

During initial training experiments, we noticed a tendency of the model to repeat the same gloss for many different word embeddings. We speculate that repeating such 'most popular' glosses might give the model higher chances of matching frequently occurring words or phrases in the dataset.

In the vocabulary size optimization experiment, we used the GRU model with 2 layers, hidden dimension 768, and 30 tokens limit during training. Table 1 summarizes our results on the trial set. We selected vocabulary size *1500* as it has the highest BLEU scores, relatively good MoverScore and the most promising glosses during manual evaluation. 1500 tokens are 8.3%, the result is close to 10%, confirming hypothesis 1 experimentally.

4.2 Model size scaling

Recent trends in deep learning suggest that bigger models increase performance on most tasks (Brown et al., 2020; Rae et al., 2021). However, the focus in these cited papers is given to Transformer (Vaswani et al., 2017), Convolutional (ConvNets) or Residual networks (ResNets). Classical recurrent neural networks (RNN) such as GRU or LSTM have been left out of the mainstream investigation of scaling effects. In the following experiments, we show that scaling RNNs also gives similar positive effects as for other network architectures. Our approach could be formulated with the following hypothesis:

Hypothesis 2 (H2): *Scaling RNNs in depth or width improves their performance.*

We summarize our experiments in tables 2, 3 and 4. The results tentatively confirm hypothesis 2. We observe that no matter how one scales the model, in width (higher hidden dimension) or depth (more layers), the performance does increase in both cases. Of course, these results are only for relatively small models fitting into our compute capacity (trained using a single Nvidia V100 GPU).

Layers	MoverScore	S-BLEU	L-BLEU
1	0.11458	0.02564	0.02561
2	0.11312	0.02427	0.02426
4	0.12454	0.02548	0.02548

Table 2: Scaling GRU model layers with fixed hidden size: 3072 dim.

Hidden	MoverScore	S-BLEU	L-BLEU
512	0.10851	0.02439	0.02437
1024	0.10880	0.02342	0.02341
3072	0.11312	0.02427	0.02426
4096	0.11071	0.02453	0.02450

Table 3: Scaling hidden dimensions for 2 layer GRU model.

4.3 Double descent

Classical machine learning theory says that increasing the model size or training time beyond some optimum, while keeping the amount of data constant, will eventually lead to the model overfitting. (i.e., bigger models would give worse performance than optimally sized smaller models). Recently, a new effect was discovered (Nakkiran et al., 2019) which contradicts, or amends, this traditional wisdom. The double descent effect states that increasing the model size (i.e., model-wise double descent) or compute resources invested into training (i.e., epoch-wise double descent) indeed leads to overfitting at first, but further increasing the size of the model or the training time can, at some critical point, reverse the trend, so that performance starts increasing again.

During our model scaling experiments we aimed

Hidden	MoverScore	S-BLEU	L-BLEU
1024	0.07284	0.02018	0.02014
2048	0.11915	0.02430	0.02427
3072	0.12454	0.02548	0.02548

Table 4: Scaling hidden dimensions for 4 layer GRU model.

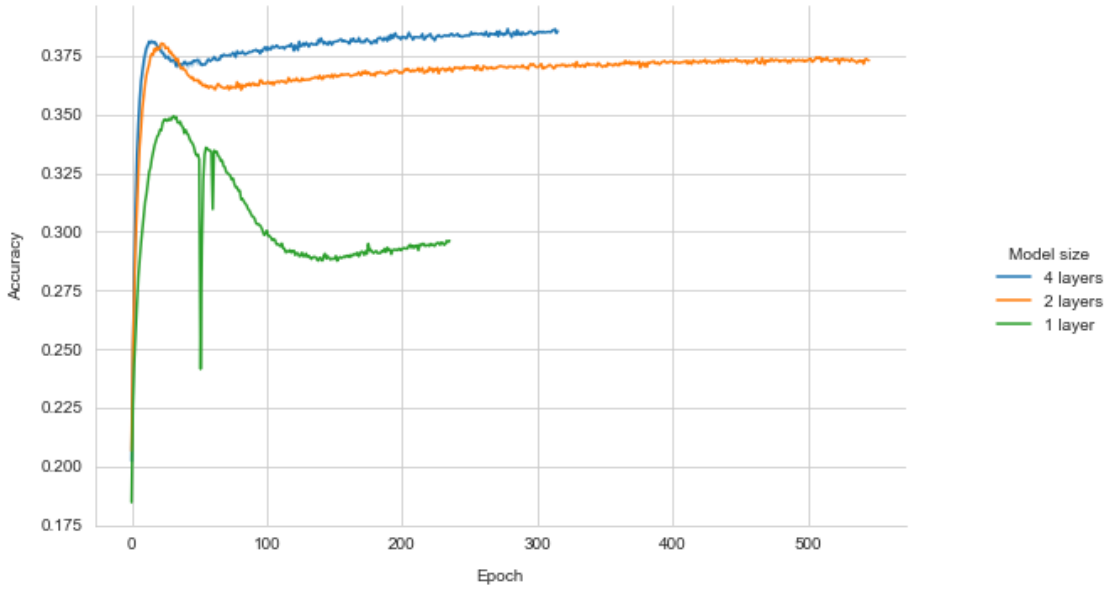

Figure 1: Accuracy: Correctly predicted word tokens, assuming that all previous word tokens were correct. The scores are calculated on the development (validation) dataset. In all cases, the hidden layer size is 3072 dim.

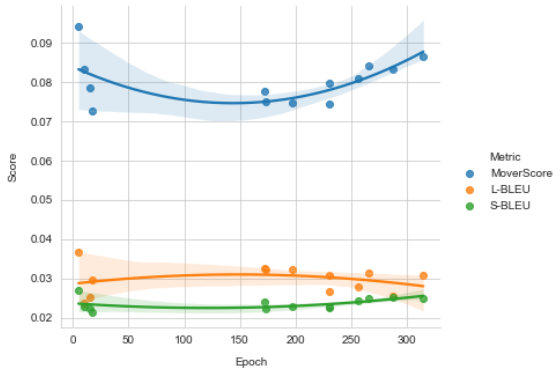

Figure 2: Automatic metric change during the 4-layer model training. The scores are calculated on the trial dataset.

to replicate the double descent effect and bring the model quality to a new level after initial overfitting. Since our compute resources were limited and we could not scale our model size endlessly, we investigate the following hypothesis:

Hypothesis 3 (H3): *Training the GRU model longer leads to an epoch-wise double descent effect.*

For our experiments, besides automatic evaluation metrics for the defmod task we introduced also an accuracy score.

Definition 1 (Accuracy): *Percent of correctly predicted tokens when all previous input tokens are correct.*

In figure 1 are 3 plots for 1-, 2- and 4-layer GRU models. The 4-layer model shows a clear epoch-wise double descent effect. We can also observe that, as previously demonstrated for other kinds of models, the effect occurs only when the model size is big enough relative to the training set. The 1- and 2-layer models are apparently too small for this training set and the task complexity.

Figure 2 is for the same 4-layer model, but in this case plotting scores on the metrics used for the CODWOE defmod task. We can see some correlation with the accuracy plot in figure 1, but these metrics seem to be less sensitive overall.

In the table 5 we illustrate the continuing gloss quality improvement according to human judgement after the first accuracy spike in the automatic metrics (epoch 5). Glosses become semantically closer to the original word. Hence, we conclude that hypothesis 3 is empirically confirmed.

5 Results

Our team "emukans" and "guntis" placed in the middle of the final ranking table. However, if we inspect the scores in figure 3, we see that our solution (a green line) does outperform others in some metrics for some languages (i.e., top score for French, 2nd for Spanish, 3rd for Russian).

Analysing our submission results, we noticed that MoverScore can give even negative scores and is quite variable from one example to another. The score is generally very low if the generated gloss length differs substantially (either too long or too

Word	Ground-truth	Epoch	Predicted	MoverScore
scraggy	Lean or thin, scrawny.	5	A slightly used to slightly.	0.11885
		193	Adorned with one or more gauntlets	0.06659
		315	Ase, slender, thin .	0.12597
coal	A glowing or charred piece of coal, wood, or other solid fuel.	5	slightly; to slightly.	0.00863
		193	A blust or furnished vehicle .	0.17182
		315	supply with energy, especially of a person's size.	0.10929
beautiful	Pleasant; clear.	5	having been (a person); to suggest or despons.	0.11287
		193	sufficient attention or thought, especially concerning the avoidance of harm.	0.03470
		315	suitable or proper; extraordinary; epic.	0.19444
thirsty	Craving something.	5	having been used to suggest or slightly.	0.04727
		193	Causing by a sensation of alcohol or narcotics.	-0.02330
		315	Causing by anger or excitement.	0.05691

Table 5: The evolution of gloss prediction during training. **N.B.** The word in column one is informational only, it was not available in the train/dev datasets and was not used during training nor prediction.

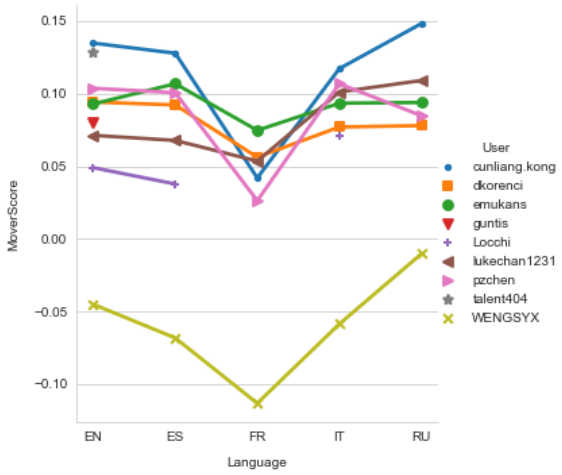

Figure 3: Best MoverScore results for all participants in all languages.

short) from the length of the ground-truth gloss, irrespective of whether a human can perceive some semantic alignment between the two.

Analysing the available data, we see that many of the glosses are relatively short: up to 20 tokens (but there are also very long examples). We conjecture that one strategy for increasing MoverScore might be to simply limit all generated glosses to 20 tokens or less.

6 Conclusion

In this competition, we tried a classical recurrent neural network approach for the CODWOE definition modeling task, and obtained positive results.

Several topics require deeper investigation. A good metric for automatically measuring how semantically close are two sentences is still an unsolved problem. MoverScore is still too far from human judgement. Taking into account even the best scores for the definition modeling task, the task is still in very early stages, and models that are trained only on the provided data cannot generate any practically useful outputs. This could possibly be addressed with much larger training datasets, or by allowing the use of external data (or of large pretrained language models). In general, it seems that a word semantic could not be represented using a single vector. The task requires more context to capture the semantics. Maybe the task could be changed to generating a gloss for a set of synonyms or semantically close words.

Acknowledgements

This research was supported by the Latvian Council of Science, project No. lzp-2021/1-0479, and by the H2020 project SELMA (under grant agreement No. 957017).

References

Tom Brown, 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. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, Doha, Qatar. Association for Computational Linguistics.

Daniel Duckworth, Arvind Neelakantan, Ben Goodrich, Lukasz Kaiser, and Samy Bengio. 2020. Parallel scheduled sampling.

Mitchell A Gordon, Kevin Duh, and Jared Kaplan. 2021. Data and parameter scaling laws for neural machine translation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5915–5922, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9:1735–80.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *CoRR*, abs/2001.08361.

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *International Conference on Learning Representations*.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing.

Timothee Mickus, Denis Paperno, Mathieu Constant, and Kees van Deemter. 2022. SemEval-2022 Task 1: Codwoe – comparing dictionaries and word embeddings. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Preetum Nakkiran, Gal Kaplun, Yamini Bansal, Tristan Yang, Boaz Barak, and Ilya Sutskever. 2019. Deep double descent: Where bigger models and more data hurt. *CoRR*, abs/1912.02292.

Lutz Prechelt. 2012. *Early Stopping — But When?*, pages 53–67. Springer Berlin Heidelberg, Berlin, Heidelberg.

Jack W. Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, H. Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, Eliza Rutherford, Tom Hennigan, Jacob Menick, Albin Cassirer, Richard Powell, George van den Driessche, Lisa Anne Hendricks, Maribeth Rauh, Po-Sen Huang, Amelia Glaese, Johannes Welbl, Sumanth Dathathri, Saffron Huang, Jonathan Uesato, John Mellor, Irina Higgins, Antonia Creswell, Nat McAleese, Amy Wu, Erich Elsen, Siddhant M. Jayakumar, Elena Buchatskaya, David Budden, Esme Sutherland, Karen Simonyan, Michela Paganini, Laurent Sifre, Lena Martens, Xiang Lorraine Li, Adhiguna Kuncoro, Aida Nematzadeh, Elena Gribovskaya, Domenic Donato, Angeliki Lazaridou, Arthur Mensch, Jean-Baptiste Lespiau, Maria Tsimpoukelli, Nikolai Grigorev, Doug Fritz, Thibault Sottiaux, Mantas Pajarskas, Toby Pohlen, Zhitao Gong, Daniel Toyama, Cyprien de Masson d'Autume, Yujia Li, Tayfun Terzi, Vladimir Mikulik, Igor Babuschkin, Aidan Clark, Diego de Las Casas, Aurelia Guy, Chris Jones, James Bradbury, Matthew Johnson, Blake A. Hechtman, Laura Weidinger, Iason Gabriel, William S. Isaac, Edward Lockhart, Simon Osindero, Laura Rimell, Chris Dyer, Oriol Vinyals, Kareem Ayoub, Jeff Stanway, Lorrayne Bennett, Demis Hassabis, Koray Kavukcuoglu, and Geoffrey Irving. 2021. Scaling language models: Methods, analysis & insights from training gopher. *CoRR*, abs/2112.11446.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Fei Liu Yang Gao Christian M. Meyer Steffen Eger Wei Zhao, Maxime Peyrard. 2019. Moverscore: Text generation evaluating with contextualized embeddings and earth mover distance. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, Hong Kong, China. Association for Computational Linguistics.

Ronald J. Williams and David Zipser. 1989. A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, 1(2):270–280.