Vec2Sent: Probing Sentence Embeddings with Natural Language Generation

Martin Kerscher and **Steffen Eger**
Natural Language Learning Group (NLLG)
Technische Universität Darmstadt
martin.kerscher@protonmail.com, eger@aiphes.tu-darmstadt.de

Abstract

We unveil the language encoded in sentence embeddings by conditionally generating from them. We perceive of this as a new unsupervised probing task and show that it correlates well with downstream task performance. We also illustrate how the language generated from different encoders differs. We apply our approach to generate sentence analogies from sentence embeddings.

1 Introduction

Generalizing the concept of word embeddings to sentence level, sentence embeddings (a.k.a. sentence encoders) are ubiquitous in NLP as features in downstream classification tasks and in semantic similarity and retrieval applications (Kiros et al., 2015; Conneau et al., 2017). *Probing sentence encoders* for the linguistic information signals they contain has likewise become an important field of research, as this allows to introspect otherwise black-box representations (Adi et al., 2017; Conneau et al., 2018). The idea behind probing tasks is to query representations for certain kinds of linguistic information such as the dependency tree depth of an encoded sentence. There are a variety of problems surrounding current probing task specifications: (i) probing tasks need to be manually construed, which brings with it a certain degree of arbitrariness and incompleteness; (ii) most probing tasks require labeled datasets or trained classifiers such as dependency parsers for linguistic processing—however, these may be unavailable for many low-resource languages or available only to a limited degree; (iii) it is not entirely clear how probing tasks have to be designed, e.g., how much training data they require and which classifier to use for probing (Eger et al., 2020); (iv) Ravichander et al. (2020) also argue that standard probing tasks do not outline the information signals a classifier actually uses for making predictions.

Our contribution is to design an alternative, more direct introspection of sentence embeddings, namely, through conditional natural language generation, which we call "vec2sent" (V2S). By retrieving and (manually) investigating the discrete output obtained from a dense vector representation, linguistic properties of the embedding may be 'directly' unveiled: e.g., we expect that a word-order insensitive model would have a comparatively hard time in restoring the correct syntax of an encoded sentence. V2S requires no labeled data, making it applicable to any language that has at least several ten thousands of written sentences available—e.g., it is particularly suitable for multilingual probing (Krasnowska-Kieraś and Wróblewska, 2019; Eger et al., 2020). Since V2S makes the opaque space $\mathbb{R}^d$ observable, it may also reveal intriguing properties of how encoders encode text (cf. Table 2), without having to 'guess' relevant probing tasks.[1]

2 Approach

For an input sentence $\vec{x} \in V^*$ (a sequence of tokens over a vocabulary V), consider its sentence representation $\mathbf{x} \in \mathbb{R}^d$ induced by some sentence embedding model E. We consider a decoder D which takes $\mathbf{x}$ as input and produces natural language text as output. We train D to reconstruct $\vec{x}$ from $\mathbf{x}$. For simplicity, we design D as a recurrent neural network (RNN) with LSTM cells, rather than as a more

[1] Our code is available at https://github.com/maruker/vec2sent.

Proceedings of the 28th International Conference on Computational Linguistics, pages 1729–1736
Barcelona, Spain (Online), December 8-13, 2020

recent model class like Transformers. At each time step i in the RNN, the goal is to predict the next word given the previously generated word y_{i-1} as input as well as the hidden state vector $\mathbf{h}_i$ which summarizes all past observations. To implement generation conditional upon the sentence embedding $\mathbf{x}$, we concatenate the embedding $\mathbf{x}$ to each input embedding ($\mathbf{y}_{i-1}$). Figure 1 illustrates our approach.

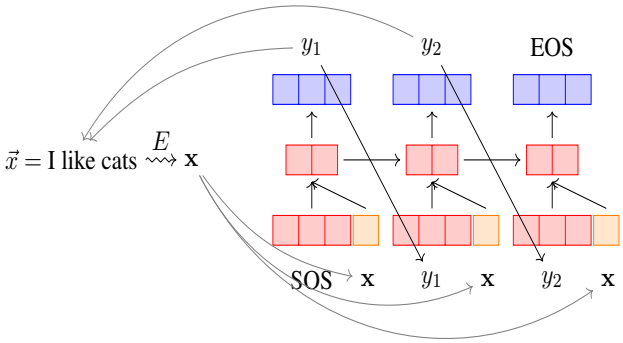

Figure 1: Schematic illustration of our approach: A pre-trained sentence encoder E maps the input $\vec{x}$ to a vector $\mathbf{x}$. This vector is fed into an RNN language model. The output $\vec{y} = (y_1, y_2, \ldots)$ generated by the conditional RNN is finally compared to the original input sentence.

In general, V2S translates from a continuous vector space to a discrete language space, while an embedding model performs the inverse operation:

$$\mathbf{x} \in \mathbb{R}^d \quad \underset{E}{\overset{\text{V2S}}{\rightleftarrows}} \quad \vec{x} \in V^*$$

The continuous space in which $\mathbf{x}$ lies is 'opaque' (to humans) and requires probing tasks for introspection. The discrete space in which $\vec{x}$ lies can easily be introspected by humans (at least at small scale). An interesting application of the duality of E and V2S is that one can interpolate any two vectors $\mathbf{x}$ and $\mathbf{y}$ via $\mathbf{z}(\alpha) = \alpha \cdot \mathbf{x} + (1 - \alpha)\mathbf{y}$—using V2S, one can then decode $\mathbf{z}(\alpha)$ to derive its discrete representation for all sentences $\vec{z}(\alpha)$ "between" $\vec{x}$ and $\vec{y}$. Analogously, we could decode sentence representations to find analogies akin the famous king-woman+man equation for word embeddings (Diallo et al., 2019).

Diagnostic tests The output $\vec{y}$ of the language model can be compared to the original input $\vec{x}$. Intuitively, the closer $\vec{y}$ is to $\vec{x}$, the better. While manual introspection of the results may yield important insights into the deficiencies (or capabilities) of an encoder, human introspection may both be unfeasible at large scale and subjective. To this end, we define several diagnostic tests:

- `Id`$(\vec{x}, \vec{y})$: the fraction of sentences for which the condition $\vec{y} = \vec{x}$ holds.
- `PERM`$(\vec{x}, \vec{y})$: the fraction of sentences where $\vec{y}$ is a permutation of $\vec{x}$, i.e., whether $\vec{y}$ can be obtained by permuting the words in $\vec{x}$.
- `Id/PERM`: the division of `Id` by `PERM`.
- `BLEU`$(\vec{x}, \vec{y})$: measures the n-gram overlap between $\vec{x}$ and $\vec{y}$. We report average BLEU across our sample of sentences.
- `Mover`$(\vec{x}, \vec{y})$: We use MoverScore (Zhao et al., 2019), a recent soft evaluation metric based on BERT embeddings (Devlin et al., 2018), which has shown much better correlations with humans than BLEU.

3 Experiments

Sentence Encoders We consider two types of encoders, **non-parametric methods** which combine word embeddings in elementary ways, without training; and **parametric methods**, which tune parameters on top of word embeddings. As non-parametric methods, we consider: (i) average word embeddings as a popular baseline, (ii) GEM (Yang et al., 2018), a weighted averaging model, (iii) hierarchical embeddings

(Shen et al., 2018), an order-sensitive model where a max-pooling operation is applied to averages of word 3-gram embeddings in a sentence, (iv) the concatenation of average, hierarchical and max pooling (Rücklé et al., 2018), and (v) sent2vec (Pagliardini et al., 2018), a compositional word n-gram model. For (i)-(iv) we use BPEmb subword embeddings (Heinzerling and Strube, 2018) as token representations.

As parametric methods, we consider: InferSent (Conneau et al., 2017), which induces a sentence representation by learning a semantic entailment relationship between two sentences; QuickThought (Logeswaran and Lee, 2018) which reframes the popular SkipThought model (Kiros et al., 2015) in a classification context; LASER (Artetxe and Schwenk, 2018) derived from massively multilingual machine translation models; and sentence BERT (SBERT) (Reimers and Gurevych, 2019), which fine-tunes BERT representations on SNLI and then averages fine-tuned token embeddings to obtain a sentence representation. The encoders and their sizes are listed in Table 1.

Encoder	Size	Rank DS	Rank V2S
Avg	300	7	5
GEM	300	9	9
Hier	300	8	6
Avg+Max+Hier	900	6	4
Sent2Vec	700	5	8
InferSent	4096	3	1
QuickThought	4800	2	2
LASER	1024	4	3
SBERT	1024	1	7

Table 1: Encoders, their dimensionalities, and rank of encoders according to downstream (DS) and V2S diagnostic tasks. Ranks for DS are after averaging across all tasks; ranks for V2S are from the `Id` diagnostic test.

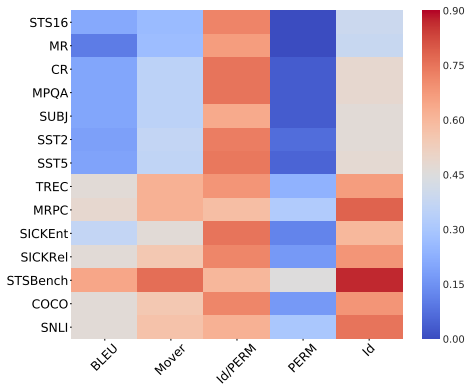

Figure 2: Spearman rank correlation between-downstream tasks from SentEval and V2S diagnostic tests.

Setup As a decoder D, we use the Mixture of Softmax RNN language model (MOS) proposed in Yang et al. (2018), which addresses the 'softmax bottleneck' in standard softmax language models and has achieved excellent results on benchmark tasks. For each encoder, we train on 1.4m sentences from News Crawl 2007[2] with the objective to retrieve the original sentence $\vec{x}$ from $\mathbf{x} = E(\vec{x})$. We evaluate performance on 10k sentences from News Crawl 2008. For speed reasons, we restrict sentence length to at most 15 tokens. Our RNN has 3 hidden layers with 1024 hidden units in each. In §3.3, we verify that the design choice of the decoder (e.g., MOS architecture vs. a simple RNN) plays a marginal role for the conclusions of V2S in our experiments.

In the sequel, we test whether V2S can predict downstream task performance (§3.1) before introspecting the language generated by different encoders (§3.2).

3.1 Correlation with Downstream Tasks

In Figure 2, we plot Spearman rank correlations (as in Conneau and Kiela (2018)) between our V2S statistics and the 14 downstream tasks for standardized comparison of sentence encoders from SentEval (Conneau and Kiela, 2018); full results are given in the appendix. The correlations indicate whether rankings of encoders induced by the probing tasks transfer to downstream tasks: i.e., to which degree it holds that $A \succ_{\text{Probing}} B$ implies that $A \succ_{\text{Downstream}} B$, for encoders A and B. The downstream tasks and V2S have no training data overlap.

Results We observe the following: (i) `PERM` is not strongly (but in general positively) correlated with downstream task performance. This means that an encoder which can better identify all the words of $\vec{x}$ from $\mathbf{x}$ has (only) a small chance of performing better in a downstream task—in fact, Avg has the best performance for `PERM` but typically performs badly in downstream tasks. (ii) `Mover` beats `BLEU`.

[2]http://www.statmt.org/wmt14/training-monolingual-news-crawl/news.2007.en.shuffled.gz

This is as expected, as `Mover` can better take paraphrasing and lexical variation into account. (iii) More surprising is that `Id` beats both of them, even though `Id` is one of the simplest possible metrics: it only asks whether $\vec{x}$ could exactly be reproduced from $\mathbf{x}$. (iv) The best correlation with downstream tasks in our context has `Id/PERM`. This means that an encoder performs better on downstream tasks if it satisfies two conditions: it can correctly identify all words in $\vec{x}$ *and* place them in correct word order.

Comparison with (other) probing tasks We contrast with the other 10 probing tasks defined in SentEval. Compared to the likewise unsupervised Word Content (WC) tasks—which probes for specific lexical words stored in the representations—`Id` and `Id/PERM` perform much better. In fact, WC has lowest predictiveness in our experiments.[3] For instance, the average correlation with downstream tasks of WC is 0.23 and the lowest correlation is -0.09, while `Id` has an average correlation of 0.59 and a minimum of 0.38. `Id/PERM` has an average correlation of 0.69 and a minimum of 0.58. Overall, `Id/PERM` is a better predictor of downstream task performance than 7 out of 10 probing task from SentEval according to the 'min' category. It is beaten only by the three syntactic tasks TreeDepth, TopConstituents and BigramShift. When we remove SBERT as an encoder, `Id` has the best predictiveness according to 'min' among all probing tasks and diagnostic tests. Altogether, this shows that statistics based on V2S can be excellent predictors for downstream task performance for our class of sentence encoders.

It is unclear why SBERT performs rather badly according to most V2S statistics but has strong downstream task performance; cf. the rank statistics in Table 1.[4] Possibly, reconstructing text from SBERT representations is more difficult due to the contextualized nature of the embeddings and the SNLI fine-tuning objective (Wang and Kuo, 2020). Note, however, that `Id/Perm` ranks SBERT on place 2nd, much closer to its actual downstream task performance, where it is overall the best encoder.

3.2 Qualitative Analysis

Input	" the point is , " stoller adds , " i 'm not sure .	$a : b :: \mathbf{z} : c$	$a : b :: \mathbf{z} : c$
LASER	" the point is , "inger adds , " i 'm not sure .	she visits italy	she is a nurse
QT	" the point is , " i 'm not sure , " spector adds .	she visits france	she is a doctor
InferSent	the point is , bernstein adds , " i 'm not sure . "	rose visits italy	she is a doctor
Avg	" i 'm sure , it is the point , " stoller adds . "	rate is	a.i.
SBERT	" well , i don 't know , it 's my question , " steinbre	she travels spain	she is medicine

Table 2: Left: Input sentence $\vec{x}$, encoders and their sample reproductions $\vec{y}$ from vec2sent. Right: Sentence level analogies $a : b :: z : c$ for the triples (a = he visits italy, b = he eats pizza, c = she drinks wine) on the left, and (a = he is a doctor, b = his name is robert, c = her name is julia) on the right. The notation $a : b :: z : c$ means "a is to b as z to c". We use V2S to induce the missing sentence z in the analogy.

Table 2 gives a sample input sentence $\vec{x}$ and the content retrieved from its embedding $\mathbf{x}$, for various encoders. We see that LASER and InferSent embeddings almost completely contain the original sentence, in correct order; only the proper name ("Stoller") and punctuation are not properly decoded. The Avg encoder gets almost all words correctly (except for missing a negation) but reverses their order, thus completely changing the text's meaning. The decoded sentence underlying the SBERT representation is furthest away from the surface form of the input, but semantically closer than, e.g., the Avg embedding.

This is a vivid illustration why, e.g., the WC probing task may have little predictive power for downstream applications (contrary to what has been claimed in some previous research), as it may be better to represent the meaning and logical structure of a text rather than its surface form. We note that our examples are to some degree representative of the encoders. For example, among all encoders, Avg

[3]While this is in contrast with Conneau et al. (2018), who report best predictiveness of WC, it is more congruent with Perone et al. (2018) and Eger et al. (2019), who report low discrimination capabilities of WC. In the end, the choice of encoders may also play an important role in this context.

[4]We remark that Lin and Smith (2019) also report rather low performance for BERT based encoders according to the nearest neighbor overlap probing task.

embeddings most frequently permute the word order of the input and we observe absence of correctly restored lexical information in many examples for SBERT.

Next, we explore whether we can use V2S to unveil sentence level semantic analogies. For three query sentences with representations $\mathbf{r}, \mathbf{s}, \mathbf{v}$ we compute the vector $\mathbf{u} := \mathbf{r} - \mathbf{s} + \mathbf{v}$ (inspired by "king-woman+man") and give $\mathbf{u}$ as input to V2S. Illustrative results are shown in Table 2 (right). We find that Avg performs poorly and cannot solve our semantic analogies. QT, SBERT, and LASER perform much better. LASER appears to encode gender biases here. SBERT, again, seemingly stores less surface level information. Note that it is in principle also possible to look for nearest neighbors of $\mathbf{u}$ to check for semantic analogies; however, nearest neighbors depend on the sample of sentences available in the corpus—V2S can be seen as a more direct form of introspection.

3.3 Stability of the analysis

Finally, we test whether the obtained results are stable under a change of architecture (replacing the MOS language model with a simple LSTM language model—which in addition maps the embedding in the initial hidden state of the RNN via a linear transformation rather than concatenating to word embeddings—and identical training sizes) and training size for the language model (200k sentences instead of 1.4m for MOS). Table 3 shows the results. We find that all statistics are quite stable ($\rho \geq .80$ for all diagnostic tests) and do not seem to vary much along the investigated two dimensions. This is a reassuring result, as it indicates that our results are not an artefact of any of these two choices.

	Low-Res. MOS	Simple LSTM
BLEU	.90	.87
MOVER	.97	.95
ID	.92	.90
ID/PERM	.90	.82
PERM	.88	.80

Table 3: Spearman rank correlation between "high-resource" MOS setting with 1.4m training sentences and (a) low-resource setting with same architecture (b) different architecture but same training size.

4 Discussion & Conclusion

The goal of probing tasks is to make the opaque embedding space $\mathbb{R}^d$ "linguistically" observable to humans. A plethora of different probing tasks have been suggested (Linzen et al., 2016; Shi et al., 2016; Adi et al., 2017), which Conneau et al. (2018) classify into *surface*, *syntactic*, and *semantic* probing tasks. More recently, multilingual extensions of probing tasks (Krasnowska-Kieraś and Wróblewska, 2019; Sahin et al., 2019; Eger et al., 2020) have been considered, as well as word level probing for especially contextualized representations (Tenney et al., 2019; Liu et al., 2019). In the latter context, Hewitt and Manning (2019) discover an intriguing structural property of BERT, *viz.*, to contain whole syntax trees in its representations, possibly, as we show, at the expense of lexical information (at least SBERT).

We investigated V2S as an alternative, direct way of probing sentence encoders. We showed that V2S may be a good predictor of downstream task performance and, in particular, that one of two simple diagnostic tests had good predictive performance in all scenarios: whether an encoder can exactly retrieve the underlying sentence from an embedding (Id) and whether the fraction of exactly retrieved sentences among all retrieved sentences that are permutations of the input sentence is high. Thus, we recommend to report both of these diagnostic tests for V2S. We also showed that V2S allows to directly introspect for certain structural properties of the embedding space, e.g., sentence-level semantic analogies.

V2S is an unsupervised probing task and, as such, we believe that a particularly interesting use case will be for low-resource languages and scenarios, for which our experiments suggests that it will be a much better predictor of downstream task performance than the equally unsupervised (and most closely related) WC probing task.

References

Yossi Adi, Einat Kermany, Yonatan Belinkov, Ofer Lavi, and Yoav Goldberg. 2017. Fine-grained analysis of sentence embeddings using auxiliary prediction tasks. *CoRR*, abs/1608.04207.

Mikel Artetxe and Holger Schwenk. 2018. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *CoRR*, abs/1812.10464.

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

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680, Copenhagen, Denmark, September. Association for Computational Linguistics.

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

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

Aïssatou Diallo, Markus Zopf, and Johannes Fürnkranz. 2019. Learning analogy-preserving sentence embeddings for answer selection. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 910–919, Hong Kong, China, November. Association for Computational Linguistics.

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

Steffen Eger, Johannes Daxenberger, and Iryna Gurevych. 2020. How to probe sentence embeddings in low-resource languages: On structural design choices for probing task evaluation. *CONLL*.

Benjamin Heinzerling and Michael Strube. 2018. BPEmb: Tokenization-free pre-trained subword embeddings in 275 languages. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan, May. European Language Resources Association (ELRA).

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4129–4138, Minneapolis, Minnesota, June. Association for Computational Linguistics.

Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, Richard Zemel, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. 2015. Skip-thought vectors. *CoRR*, abs/1506.06726.

Katarzyna Krasnowska-Kieraś and Alina Wróblewska. 2019. Empirical linguistic study of sentence embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5729–5739, Florence, Italy, July. Association for Computational Linguistics.

Lucy H. Lin and Noah A. Smith. 2019. Situating sentence embedders with nearest neighbor overlap. *CoRR*, abs/1909.10724.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the ability of lstms to learn syntax-sensitive dependencies. *CoRR*, abs/1611.01368.

Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. 2019. Linguistic knowledge and transferability of contextual representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1073–1094, Minneapolis, Minnesota, June. Association for Computational Linguistics.

Lajanugen Logeswaran and Honglak Lee. 2018. An efficient framework for learning sentence representations. *CoRR*, abs/1803.02893.

Matteo Pagliardini, Prakhar Gupta, and Martin Jaggi. 2018. Unsupervised learning of sentence embeddings using compositional n-gram features. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 528–540, New Orleans, LA, USA, June. Association for Computational Linguistics.

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

Abhilasha Ravichander, Yonatan Belinkov, and Eduard H. Hovy. 2020. Probing the probing paradigm: Does probing accuracy entail task relevance? *ArXiv*, abs/2005.00719.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, November.

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

Gözde Sahin, Clara Vania, Ilia Kuznetsov, and Iryna Gurevych. 2019. LINSPECTOR: Multilingual probing tasks for word representations. *CoRR*, abs/1903.09442.

Dinghan Shen, Guoyin Wang, Wenlin Wang, Martin Renqiang Min, Qinliang Su, Yizhe Zhang, Chunyuan Li, Ricardo Henao, and Lawrence Carin. 2018. Baseline needs more love: On simple word-embedding-based models and associated pooling mechanisms. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 440–450, Melbourne, Australia, July. Association for Computational Linguistics.

Xing Shi, Inkit Padhi, and Kevin Knight. 2016. Does string-based neural MT learn source syntax? In *EMNLP*, pages 1526–1534. The Association for Computational Linguistics.

Ian Tenney, Patrick Xia, Chen Berlin, Alex Wang, Adam Poliak, Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel Bowman, Dipanjan Das, and Ellie Pavlick. 2019. What do you learn from context? probing for sentence structure in contextualized word representations. In *ICLR 2019*.

Bin Wang and C-C Jay Kuo. 2020. SBERT-WK: A sentence embedding method by dissecting bert-based word models. *arXiv preprint arXiv:2002.06652*.

Zhilin Yang, Zihang Dai, Ruslan Salakhutdinov, and William W. Cohen. 2018. Breaking the softmax bottleneck: A high-rank RNN language model. In *International Conference on Learning Representations*.

Wei Zhao, Maxime Peyrard, Fei Liu, Yang Gao, Christian M. Meyer, and Steffen Eger. 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, August. Association for Computational Linguistics.

A Supplementary Material

A.1 Scores

Encoder	BLEU	MOVER	PERM	Id	Id / PERM
Average	33.51	67.62	43.22	15.18	35.12
GEM	23.76	28.58	1.44	1.21	84.03
Hier	37.9	58.61	17.29	14.97	86.58
Avg+Max+Hier	34.91	62.48	30.62	17.67	50.85
Sent2Vec	32.94	45.48	16.08	2.77	87.38
Infersent	53.54	72.01	23.21	21.19	91.29
QuickThought	48.07	70.09	22.27	20.02	89.9
LASER	52.88	73.87	33.58	18.52	96.51
SBERT	21.5	43.39	3.45	3.27	94.78

Table 4: Complete scores from the metrics evaluated on generated sentences. Experimental setup detailed in section 3

Encoder	Average	GEM	Hier	Avg+Max+Hier	Sent2Vec	Infersent	Quickthoughts	SBERT
STS	0.57	0.59	0.58	0.63	0.57	0.71	0.61	0.76
MR	72.18	63.46	68.34	72.16	75.16	79.44	82.58	84.79
CR	74.49	72.56	72.66	75.6	77.51	84.32	84.66	90.81
MPQA	74.83	73.26	74.48	76.19	87.44	89.37	89.97	90.43
SUBJ	89.47	78.24	86.53	89.16	91.58	92.64	94.8	94.47
SST2	76.39	64.52	72.32	74.68	78.91	84.57	88.08	90.83
SST5	39.1	32.71	37.65	39.1	40.68	45.79	48.64	50.09
TREC	69.6	66	64.4	69	76.4	90.6	91.4	86.6
MRPC F1	80.95	72.31	79.74	81.75	80.74	83.74	84.05	82.87
SICK E	74.28	70.25	72.97	76.05	78.77	85.59	82.73	83.05
SICK R	0.63	0.57	0.59	0.67	0.72	0.83	0.81	0.8
STSb	0.57	0.47	0.55	0.61	0.53	0.78	0.79	0.75
COCO	26.1	16.32	21.24	28.8	29.01	43.02	42.32	38.12
SNLI	63.86	52.46	62.33	67.35	59.84	84.47	78.06	83.92

Table 5: Complete scores from downstream tasks evaluated using senteval.