Second-order Document Similarity Metrics for Transformers

Jarkko Lagus
Department of Computer Science
University of Helsinki
jarkko.lagus@helsinki.fi

Niki Loppi
NVIDIA
nloppi@nvidia.com

Arto Klami
Department of Computer Science
University of Helsinki
arto.klami@helsinki.fi

Abstract

The similarity of documents represented using static word embeddings is best measured using second-order metrics accounting for the covariance of the embeddings. Transformers provide superior representations for words compared to static embeddings, but document representation and similarity evaluation are currently often done using simple mean pooling. We explain how the second-order metrics can be used also with transformers, and evaluate the value of improved metrics in this context.

1 Introduction

Many NLP models rely on pretrained representations, either static embeddings (e.g. Word2Vec by Mikolov et al. (2013)) or context-aware models like transformers (e.g. BERT by Devlin et al. (2018)) that process text sequentially but still represent each word or subword with a fixed-dimensional latent representation. These models are then fine-tuned to solve specific tasks, by continuing to train the representations while optimizing for the task performance.

Longer documents cannot be directly modeled by most transformers, but a representation can be obtained by processing them in smaller units (e.g. sentences) and then *pooling* the representations of the individual words. The pooling method is often a simple mean or max pooling, but despite the simplicity, such approaches work relatively well both with static embeddings (Wieting et al., 2015; Arora et al., 2017; Gupta et al., 2020) and transformers (Devlin et al., 2018; Reimers and Gurevych, 2019).

For instance, Torki (2018); Nikolentzos et al. (2017); Muzellec and Cuturi (2018) and Lagus et al. (2019) have shown that for static embeddings the similarities can be more accurately captured by accounting for the covariance structure of the word matrix using so-called second-order metrics. We study whether this holds also for transformers. This is not obvious upfront, since transformers process documents sequentially and hence capture some document-level information already in (sub)word representations, and directly fine-tuning the representations for accurate document comparison may enable even mean pooling to capture some of the same information the second-order representations use.

We explain how model-agnostic second-order metrics can be used with transformers, provide details for fine-tuning both full-rank (Torki, 2018) and low-rank (Mu et al., 2017; Yang et al., 2018; Lagus et al., 2019) metrics using the new pooling functions, and evaluate them in three tasks. We focus on precomputable document representations, not relying on similarity comparisons that require cross-encoding the document pairs like in the original BERT paper by Devlin et al. (2018). This makes the methods suitable for online processing and enables caching representations in a database. Second-order metrics have computational overhead in isolation, but the fine-tuning process is dominated by other parts of the model and hence the added computation time is small, even for low-rank metrics that require propagating gradients through singular-value decomposition (SVD). We show that second-order metrics improve document similarity comparison in two languages and for two transformer models, but do not help for sentence similarity.

2 Document Metrics

We denote by $A \in \mathbb{R}^{n \times d}$ a matrix that collects the d-dimensional representations of the n words occurring in the document as its rows.

The first-order metrics compress A into a d-dimensional vector, typically using the mean $a = \frac{1}{n}\mathbf{1}^T A$, where $\mathbf{1}$ is n-dimensional vector of ones, and compare the documents e.g. by cosine similarity $S_{cos}(a, b) = (a \cdot b)(\|a\|\|b\|)^{-1}$ where b denotes the mean vector for another document B (typically with different n). See Arora et al. (2017) for details (e.g. weighting) for static embeddings and Reimers and Gurevych (2019) for use with transformers.

The second-order metrics are based on covariance-like products $A^TA \in \mathbb{R}^{d\times d}$ capturing both variance and correlation between the representation dimensions. Several slightly different metrics have been proposed: Torki (2018) vectorized the covariance and combined it with the mean pooling, Lagus et al. (2019) derived a second-order metric from pair-wise cosine similarity of all word pairs in the two documents, and Muzellec and Cuturi (2018) used Wasserstein metric to compare second-order representations. Next, we describe the specific metric of Lagus et al. (2019) that supports also low-rank computation proposed for computational reasons but note that the other metrics can be implemented as minor modifications.

The second-order metrics compare A^TA and B^TB. A metric normalized to range $[-1, 1]$ can be conveniently expressed as $S_F(A^TA, B^TB)$ using the general similarity measure

$$S_F(X, Y) = \frac{\langle X, Y\rangle_{\mathrm{F}}}{\|X\|_{\mathrm{F}}\|Y\|_{\mathrm{F}}}, \tag{1}$$

where $\langle X, Y\rangle_{\mathrm{F}} = \mathrm{Tr}\left(X^TY\right)$ is the Frobenius inner product and $\|X\|_{\mathrm{F}} = \sqrt{\langle X, X\rangle_{\mathrm{F}}}$.

The obvious drawback of the metric is that A^TA has $O(d^2)$ elements, compared to $O(d)$ of the mean representation. Lagus et al. (2019) showed that forming the covariance matrix can be avoided by computing SVD of A: If $A = U\Sigma V^T$ then $A^TA = V\Sigma U^TU\Sigma V^T = V\Sigma^2V^T$, and the trace term can be evaluated using the SVDs of the two document matrices. They also demonstrated empirically that a low-rank approximation for the metric, using $A_k = \Sigma_k V_k^T \approx A$ with $k \ll \min(n, d)$ components has a regularizing effect while reducing computation time compared to full-rank matrices.

3 Fine-tuning Second-order Metrics

Fine-tuning of transformers is done end-to-end for a model that combines four parts: the embedding model $E(\cdot)$, a pooling function $P(\cdot)$, the similarity metric $S(\cdot,\cdot)$ and some eventual loss function $L(\cdot,\cdot)$ that compares the similarities with ground truths. If we denote by $A = E(d_A)$ the matrix formed by the embedding model processing a text sequence and by $\tilde{S}$ the ground truth similarity, the objective to be trained with respect to parameters of $E(\cdot)$, starting with the pretrained values, is

$$L(S(P(E(d_A)), P(E(d_B))), \tilde{S}(d_A, d_B)).$$

3.1 Pooling for Second-order Metrics

The current modeling pipelines building on mean pooling $P_{mean}(A) = \frac{1}{n}\mathbf{1}^TA$ can be re-used with second-order metrics, by re-writing the metrics as a combination of a generalized pooling function and a distance measure. For this, we assume $P(A)$ to be any function that transforms A to a fixed-dimensional representation (vector or matrix). Then we can define the new pooling functions

$$P_{cov}(A) = A^TA \quad \text{and} \quad P_{svd}(A, k) = \sqrt{\Sigma_k}V_k^T,$$

where Σ_k and V_k refer to matrices that retain the singular values and vectors corresponding to k largest values. In the experiments, we also use additional method derived from $P_{cov}(\cdot)$ named, $P_{cov}(\cdot, k)$ where the dimensionality reduction to k dimensions is done after the fine-tuning in the prediction phase. Then the three metrics for document comparison can be written as

$$\begin{aligned} \text{mean} &: S_{cos}(P_{mean}(A), P_{mean}(B)), \\ \text{full-rank} &: S_F(P_{cov}(A), P_{cov}(B)), \\ \text{low-rank} &: S_F(P_{svd}(A, k), P_{svd}(B, k)). \end{aligned}$$

This unified formulation makes it easy to implement the second-order metrics as part of standard processing pipelines. The implementation of the second-order metrics, compatible with the *sentence-transformers* library by Reimers and Gurevych (2019), is made available on GitHub[1].

The size of $P_{cov}(\cdot)$ is quadratic in d and hence can be large, which is generally considered a challenge in the case of static embeddings. In the context of transformers, however, this is insignificant since we are anyway fine-tuning a typically very large model. The practical computation is hence largely dominated by the other parts of the pipeline (see Section 4 for empirical validation).

3.2 Differentiable SVD

Fine-tuning for full-rank second-order metrics does not require special treatment, but for low-rank metrics we need to propagate gradients through the singular value decomposition used for approximating A^TA as described in Section 2. Deep learning frameworks offer differentiable SVD implementations out-of-the-box, providing multiple approximate algorithms with varying properties. While the forward pass through SVD is relatively fast and stable for all choices, the backward pass is

[1]https://github.com/jalagus/second-order-transformers

more costly and prone to instabilities arising from ill-conditioned document matrices. We found the CPU version of `torch.svd_lowrank`, implementing the algorithm of Halko et al. (2011), to be the most stable and hence use that.

Another challenge is the sign ambiguity over the matrices U and V. The pooling function $P_{svd}(\cdot, k) = \sqrt{\Sigma_k} V_k^T$ is hence not unique which causes issues with the distance computation; some components might point to opposite directions while still encoding the same information.

To address these issues we explicitly reconstruct the covariance matrix $A^T A$ to improve backward pass stability and remove the sign ambiguity. The pooling is thus implemented as $P_{svd}(A, k) = U_k \Sigma_k V_k^T$. This has no practical effect on computation speed as it is dominated by other components of the full model. Since $A^T A$ is symmetric, the U and V matrices will be identical and it is enough to save the matrix $D = \sqrt{\Sigma_k} V_k^T$ and reconstruct the full covariance inference-time using $D^T D$. The space requirement is then $O(kd)$.

The same space reduction for the representation can be obtained by fine-tuning using the full-rank covariance and computing the low-rank representation using SVD only after the training is done. This approach does not need a differentiable SVD, but as we will later show it performs worse in practice.

4 Experiments

We demonstrate second-order metrics in three example cases; sentence similarity on the STS benchmark (Cer et al., 2017) and two document similarity tasks. For all experiments, we compare the three alternative pooling methods and provide results for alternative embedding methods.

4.1 STS Benchmark: Sentence Similarity

Even though our main goal is to provide tools for longer documents, we also evaluate the methods in the STS sentence matching benchmark. The STS experiment was conducted using the *sentence-tranformers* library, by replacing only the pooling method and the distance computation according to the proposed metric. We compared three pooling operations (mean, covariance, and SVD with $k = 1$) for three transformers, BERT (Devlin et al., 2018), TinyBERT (Turc et al., 2019), and RoBERTa (Liu et al., 2019), and for completeness include also results on static GloVe embeddings (Pennington et al., 2014). The *sentence-tranformers* library handles the input tokenization and the default mean pooling serves as a baseline representing the current practice in the field.

Table 1: Sentence similarity on STS benchmark.

Model	Pooling	IC	MC	WT
BERT	$P_{cov}(\cdot)$	0.5957	0.8831	16.59
BERT	$P_{svd}(\cdot, 1)$	0.5900	0.8838	336.37
BERT	$P_{mean}(\cdot)$	0.5932	0.8775	11.67
TinyBERT	$P_{cov}(\cdot)$	0.6932	0.7975	3.29
TinyBERT	$P_{svd}(\cdot, 1)$	0.6930	0.7962	20.38
TinyBERT	$P_{mean}(\cdot)$	0.6937	0.7823	3.20
RoBERTa	$P_{cov}(\cdot)$	0.6463	0.8874	16.32
RoBERTa	$P_{svd}(\cdot, 1)$	0.6446	0.8875	312.63
RoBERTa	$P_{mean}(\cdot)$	0.6500	0.8857	12.03
GloVe	$P_{cov}(\cdot)$	0.7425	0.7425	-
GloVe	$P_{svd}(\cdot, 1)$	0.6517	0.6517	-
GloVe	$P_{mean}(\cdot)$	0.7163	0.7163	-

Table 1 shows the Pearson correlation with true similarity. Here IC denotes the initial correlation before fine-tuning and MC denotes the maximum correlation during fine-tuning. We report results on the development set, following the practice of cross-encoding with `[SEP]` tags by (Devlin et al., 2018) and using the *sentence-transformers* library for easy reproduction. We ran each configuration for 15 epochs using a batch size of 16. WT denotes the wall-time (minutes) taken to complete the entire fine-tuning process. All reported numbers are averages of five complete fine-tuning runs on different seeds, and our baseline results with $P_{mean}(\cdot)$ are in line with those reported in *sentence-transformers*.

For all embedding methods, using a second-order metric improves the similarity compared to mean pooling, but the gain is considerably smaller for the three transformers (below 1%) compared to the static embeddings (3.6%). For transformers the low-rank metrics perform identically with full-rank, whereas for GloVe using full-rank is preferred, matching the result of Lagus et al. (2019) for STS. Without fine-tuning the transformers are worse than static embeddings, highlighting the well known importance of fine-tuning them. In conclusion, second-order metrics can be used with transformers already for sentence comparison, but the gain is very small and due to the need of computing SVD on CPU the computation is slower.

4.2 Full-length document experiments

Second-order metrics are expected to be more useful for longer documents, and hence we evaluate them on two document similarity tasks. In both experiments, triplet loss (Schultz and Joachims,

Table 2: Document similarity: Finnish news

Model	Pooling	Initial Acc	Acc
FinBERT	$P_{mean}(\cdot)$	0.499	0.626
FinBERT	$P_{cov}(\cdot)$	0.325	0.663
FinBERT	$P_{cov}(\cdot, 1)$	-	0.484
FinBERT	$P_{cov}(\cdot, 5)$	-	0.591
FinBERT	$P_{svd}(\cdot, 1)$	-	0.644
FinBERT	$P_{svd}(\cdot, 5)$	-	0.644
fastText	$P_{mean}(\cdot)$	-	0.194
fastText	$P_{cov}(\cdot)$	-	0.073

Table 3: Document similarity: Patent retrieval

Model	Pooling	Initial Acc	Acc
TinyBERT	$P_{mean}(\cdot)$	0.553	0.885
TinyBERT	$P_{cov}(\cdot)$	0.606	0.901
TinyBERT	$P_{cov}(\cdot, 1)$	-	0.836
TinyBERT	$P_{cov}(\cdot, 5)$	-	0.867
TinyBERT	$P_{svd}(\cdot, 1)$	-	0.844
TinyBERT	$P_{svd}(\cdot, 5)$	-	0.879
fastText	$P_{mean}(\cdot)$	-	0.566
fastText	$P_{cov}(\cdot)$	-	0.592

2004) is used as the optimization target and hyperparameter optimization is done using grid search, performing computation on CPU because of instability of SVD computations on GPU.

Finnish news data We use data from the Finnish national broadcasting company *Yle* [2], of news articles written in easy-to-read Finnish, a morphologically rich language. We form an artificial task where each article is split into two equal-sized parts and the goal is to retrieve the correct second part (amongst the set of all second parts) using the first part as a query. For the triplet loss, we use the second half of a random document from the training set as the negative anchor.

For fine-tuning, we split the dataset of 600 triplets into 500 training samples and 100 validation samples. We use FinBERT (Virtanen et al., 2019) model and fine-tune it for 15 epochs, evaluating the final accuracy on a fresh set of 1500 samples. For baseline computations, we use Finnish fastText embeddings.

Patent data As a real document similarity comparison task, we consider a patent retrieval task. When patents are applied, multiple kinds of prior art might lead to rejection. Here we consider two types of prior art, namely X and A citations. The X citations are prior work that can alone lead to a rejection, while the A citations describe the state of the art, but are not immediate reasons for rejection. An ideal model would hence rank the X citations ahead of A citations.

Patent documents are split into two main parts, claims and description. We only use the claims part – containing approximately 2,100 character per document – that defines the exact claims of the invention, leaving out the more free-form description. For training, we split a proprietary dataset acquired from the United States Patent and Trademark Office of 1000 triplets into 800 training samples and 200 validation samples. In each triplet, the anchor is the patent document, the positive sample is any X citation of that patent and the negative sample is any A citation of that same patent. We fine-tune TinyBERT model for 15 epochs and evaluate it on a fresh set of 1000 documents. Static embeddings are used as a baseline.

[2]http://urn.fi/urn:nbn:fi:lb-2019121205

Results The results for both tasks, reported in Tables 2 and 3, are similar. The main observations are (a) transformers clearly outperform static embeddings, (b) fine-tuning for document similarity comparisons improves accuracy dramatically, and (c) the second-order metrics outperform the standard mean pooling clearly, with improvements of 3.7 and 1.6 percentage points for the two tasks ($P_{cov}(\cdot)$ vs $P_{mean}(\cdot)$). In conclusion, the second-order metrics are useful also in the case of transformers.

On both tasks, the total computation time for the low-rank models is only approximately 5% higher (not shown), due to need for fewer iterations for SVD, and hence the choice of the metric can be made purely based on the accuracy. Here the full-rank ($P_{cov}(\cdot)$) metrics slightly outperformed the low-rank ones ($P_{svd}(\cdot, k)$), but the latter may still be beneficial due to smaller representations. Finally, we see that extracting a low-rank representation from a model fine-tuned for the full-rank metric (denoted by $P_{cov}(\cdot, k)$) is worse than directly fine-tuning for the low-rank metric. As there is no notable difference in computational cost, we do not recommend doing this.

5 Conclusions

Transformers provide richer representations for text compared to static embeddings. For documents, the current practice is to use averages of the word representations for similarity comparisons, which is naive compared to the richer representations and distance metrics used for static embeddings.

We investigated the use of second-order metrics in the case of transformers, showed how they can be implemented into existing pipelines using pooling functions, and empirically demonstrated consistent improvement in similarity comparisons. The gain is smaller than with static embeddings but especially for longer documents still clear and consistent across different setups. Even though the improvement is not particularly large, the metrics are easy to use and hence we recommend practitioners to evaluate performance for both first-order and second-order metrics – at least the full-rank one that does not have computational overhead – and select the best metric on a validation data.

Acknowledgments

This work was supported by the Academy of Finland Flagship programme: Finnish Center for Artificial Intelligence FCAI. NL contributed under the NVIDIA AI Technology Center (NVAITC) Finland programme. In addition, we thank IPRally Technologies Oy for providing the data for the experiments done on patent data.

References

Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2017. A simple but tough-to-beat baseline for sentence embeddings. In *Proceedings of International Conference on Learning Representations*.

Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*.

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

Vivek Gupta, Ankit Saw, Pegah Nokhiz, Praneeth Netrapalli, Piyush Rai, and Partha Talukdar. 2020. P-SIF: Document embeddings using partition averaging. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):7863–7870.

Nathan Halko, Per-Gunnar Martinsson, and Joel A Tropp. 2011. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review*, 53(2):217–288.

Jarkko Lagus, Janne Sinkkonen, and Arto Klami. 2019. Low-rank approximations of second-order document representations. In *Proceedings of the 23rd Conference on Computational Natural Language Learning*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Jiaqi Mu, Suma Bhat, and Pramod Viswanath. 2017. Representing sentences as low-rank subspaces. *arXiv preprint arXiv:1704.05358*.

Boris Muzellec and Marco Cuturi. 2018. Generalizing point embeddings using the wasserstein space of elliptical distributions. In *Advances in Neural Information Processing Systems*, pages 10258–10269.

Giannis Nikolentzos, Polykarpos Meladianos, François Rousseau, Yannis Stavrakas, and Michalis Vazirgiannis. 2017. Multivariate Gaussian document representation from word embeddings for text categorization. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 450–455.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

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.

Matthew Schultz and Thorsten Joachims. 2004. Learning a distance metric from relative comparisons. In *Advances in neural information processing systems*, pages 41–48.

Marwan Torki. 2018. A Document Descriptor using Covariance of Word Vectors. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 527–532.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Well-read students learn better: On the importance of pre-training compact models. *arXiv preprint arXiv:1908.08962*.

Antti Virtanen, Jenna Kanerva, Rami Ilo, Jouni Luoma, Juhani Luotolahti, Tapio Salakoski, Filip Ginter, and Sampo Pyysalo. 2019. Multilingual is not enough: BERT for Finnish. *arXiv preprint arXiv:1912.07076*.

John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2015. Towards universal paraphrastic sentence embeddings. *arXiv preprint arXiv:1511.08198*.

Ziyi Yang, Chenguang Zhu, and Weizhu Chen. 2018. Parameter-free sentence embedding via orthogonal basis. *arXiv preprint arXiv:1810.00438*.