EasyChair Preprint

№ 3196

Tripod: Learning Latent Representations for Sequences

Tiberiu Boroș, Andrei Cotaie, Alexandru Meterez and Paul Ilioaica

April 18, 2020

TRIPOD: LEARNING LATENT REPRESENTATIONS FOR SEQUENCES

TIBERIU BOROS[1], ANDREI COTAIE[1], ALEXANDRU METEREZ[2] AND PAUL ILIOACA[2]

[1] *Adobe Inc., Security Coordination Center, {boros, cotaie}@adobe.com*
[2] *Intern/External Contributor, {alexandrumeterez, ipaulbeniamin}@gmail.com*

Abstract

We propose a new model for learning and extracting latent representations from sequences, which generates a tripartite representation: global style, memory-based and summary-based partitions. We show the relevance of these representations on a couple of mainstream tasks such as text similarity and natural language inference. We argue that the generic nature of this approach makes it applicable to many other tasks that involve modelling of discreet-valued sequences (time-ordered) and, with some modifications even to image and speech processing. We encourage everyone to try our opensource code and our Python3 API

Key words — Machine Learning, Latent representations, Embeddings, Natural Language Processing, Sequence, Sentences, Paragraphs.

1. Introduction

The emergence of word2vec (Mikolov et al.,2013) has enabled research in the direction of learning unsupervised models that generate "semantic"-oriented representations of words, tokens, images, speech etc. Such approaches include (but are not limited to) Bert (Devlin et al., 2018), ELMo (Peterset al., 2018), Glove (Pennington et al., 2014) as well as Paragraph2vec (Le and Mikolov, 2014) and the speech-oriented Global Style Tokens (GST) (Wanget al., 2018). Previous approaches limit their scope to language processing, whereas Code2vec (Alon et al., 2019) focuses on learning latent representations for computer code.

Our methodology combines lessons learned from several models and is designed to compute the latent representations of long sequences. Our approach is generic, in the sense that we extract latent representations for both text and code (described else-where). Thus, the same model architecture can be used for natural language sentences, code segments and, with minor modifications, for speech and images. However, our focus for the present work will be limited to input sequences composed of discrete valued tokens, such as text or code. We stress out that our goal is to generate latent representations for long sequences and not for individual tokens. To our knowledge, methods that achieve the same goal are Paragraph2vec (Le and Mikolov, 2014), Doc2vec (Gupta et al., 2016) and Code2vec (Alon et al., 2019).

In Section 2 we focus on describing our model, training procedure and the intuition behind our choice of architecture. Later on, in Section 2 we show the results of experiments conducted on text similarity and natural language inference. Finally, we bring conclusions and discuss further development plans in Section 4.

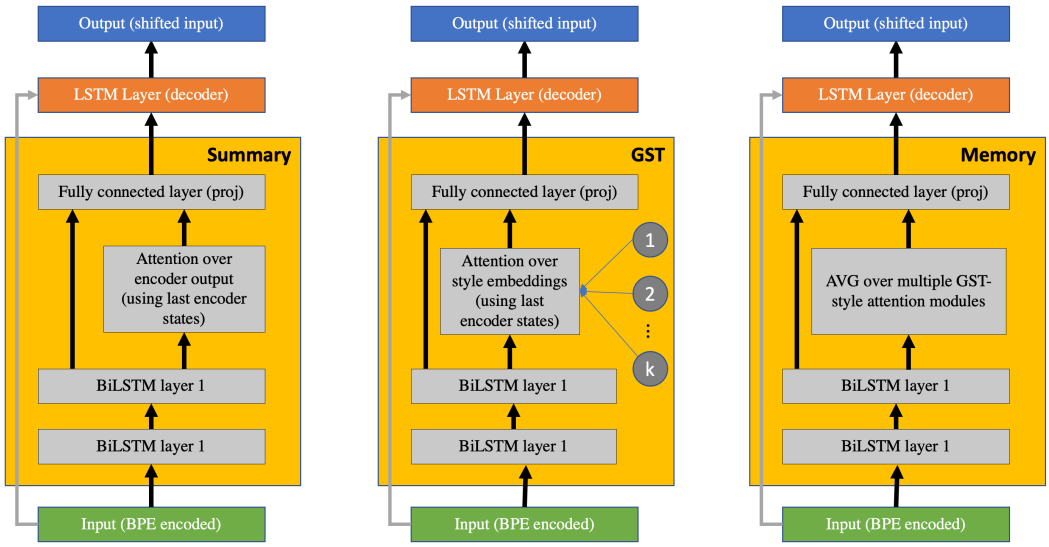

Figure 1 - Overview of Tripod's architecture

2. *Tripod model*

Several network architectures and training strategies have been previously proposed for learning how to compute latent representations in an unsupervised manner. We draw the attention that most proposed methodologies use the reconstruction of the input as the target objective, based on a type of "condensed" numeric representation, which is often the same as the latent one.

Our approach has the same training objective. However, the **difference is that we use and encoder-decoder seq2seq model** that is conditioned to reproduce the input using **three different strategies**. These strategies dictate the three latent-representation partitions: summary-based, GST-based and memory-based.

In order to generate a latent numeric representation for a sequence, we independently train three models that share a similar architecture (Figure 1) but have no parameters in common. All three methods are based on the intuition that the decoder will act as a language model (LM) that extends the standard LM estimation to incorporate "context-conditioning" (i.e. computes $P(w_k|w_1, w_2 \dots w_{k-1}, c)$, where c is a context-vector a.k.a. latent representation).

The summary-based representation uses a standard attention mechanism over the encoder states. Just like in self-attention we use the final encoder (not decoder) states in the affine transformation. Also, the context vector (Eq. 1, 2 and 3) is computed only once and the same latent representation is used to condition each decoder state in the generation phase. Because the same representation is used for every time-step, our intention is that the encoder-attention model will learn to summarize the input and retain information that is not captured by the Language Model (LM), also known as the decoder.

$$c = \sum_{i=1}^{n} h_i \, a_i \tag{1}$$

$$a_i = \frac{\exp e_i}{\sum_{k=1}^{n} \exp e_k} \tag{2}$$

$$e_k = \mathrm{MLP}\, h_k \oplus s_n^f \oplus s_n^b \tag{3}$$

In the above notations c is the context vector, n is the sequence length, h_k is the k-th output of the top-level concatenated encoder states, s_n^f, s_n^b are the k-th hidden states of the top level forward(f) and backward(b) encoder and *MLP* is a three-layer perceptron with one hidden layer ($tanh$ activation) and scalar output.

The global style-tokens (GST) representation is inspired by (Wang et al., 2018) which is used for prosody embeddings in end-to-end text-to-speech (TTS) synthesis. This approach uses a LSTM encoder that *"sees"* mel-cepstral coefficients computed from human speech samples. The last state of the encoder is used to compute an attention over a fixed number of "embeddings"[1] and, based on this attention, a conditioning vector that guides a decoder to reconstruct the input speech. Their experiments show that the global style tokens are able to capture speech-specific characteristics, such as emotion, sloppiness and speaking speed, which are transferable across utterances. In our approach the encoder is bi-directional, and we use trainable embeddings instead of cepstral coefficients.

The memory-based representation is inspired after the GST strategy, but instead of using one attention over one set of tokens we combine multiple trainable attentions with multiple groups of token groups. While the small number of tokens in GSTs forced the model to learn generic styles, this larger model is able to *"memorize"* fine-grained details about the input examples. Finally, by concatenating the context vectors from each token group and applying an affine transformation we obtained the context vector c that is used to condition the decoder. This is required in order to reduce the dimensionality of the input and computational costs.

The final latent representation is obtained by concatenating the context vectors that are independently computed by the three models.

In Section 3 we validate Tripod vectors on mainstream NLP tasks. All the experiments were conducted using a Tripod model which was pre-trained on the Wiki-103 corpus (Merity et al., 2016). Before that, we invite readers to take a look at Figure 2 which shows a graphic representation[2] of Tripod vectors[3] for the following paragraphs:

[1] Equations 1, 2 and 3 can be adapted by replacing n with the number of style-tokens and using the trainable embeddings e_i instead of the encoder outputs h_k

2 We used SVD to get 2D coordinates

3 "overall" - is the concatenated representation and the other three vectors represent the individual partitions for summary, GST and MEM

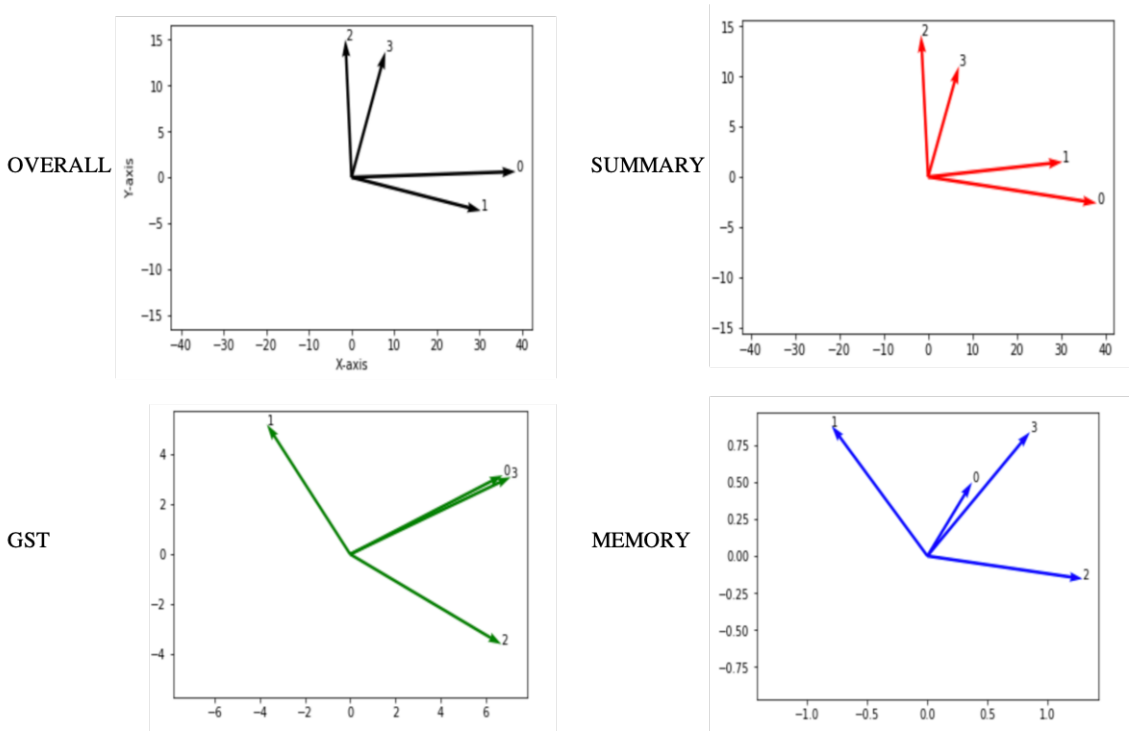

Figure 2 - Vector representation for the four sentences

"Science (from the Latin word scientia, meaning "knowledge") [1] is a systematic enterprise that builds and organizes knowledge in the form of testable explanations and predictions about the universe."

"Their contributions to mathematics, astronomy, and medicine entered and shaped Greek natural philosophy of classical antiquity, whereby formal attempts were made to provide explanations of events in the physical world based on natural causes"

"Bucharest is the capital and largest city of Romania, as well as its cultural, industrial, and financial centre. It is located in the southeast of the country"

"The city proper is administratively known as the "City", and has the same administrative level as that of a national county, being further subdivided into six sectors, each governed by a local mayor."

Note 1: In order to speed-up convergence, in the data preparation phase we used TF-IDF and SVD (20 dimensions) followed by K-Means clustering to *k* clusters, with *k* equal to the number of GST tokens. The assigned a cluster index to each data entry and was used the cluster index to guide the attention on the GST tokens.

Note 2: During the experiments in Section 3 the Tripod model was not fully converged.

Note 3: Initially, the computed representations were close to the centroids of the clusters. As the model converged, they drifted away from these centroids and the accuracy on the test datasets (SNLI and SICK – see Section 3) got better.

Note 4: Without the measures describe in (1), some of our models did not converge at all.

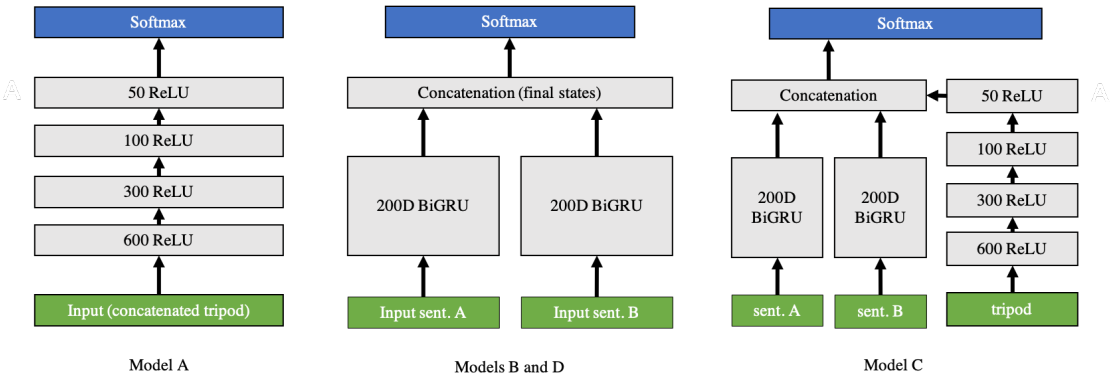

Figure 3 - The three model architectures used in our experiments

3. *Experimental validation*

We validate our approach on two datasets that require Natural Language Inference: the SICK dataset (see Section 3.1) and the SNLI dataset (see Section 3.2).

3.1. *The SICK 2014 dataset*

Sentences Involving Compositional Knowledge (SICK) dataset (Marelli et al., 204) is composed of 10K English sentence pairs. Each pair is annotated with relatedness score and entailment information, which is strictly directional (A in ENTAILMENT with B does not imply B in ENTAILMENT with A).

Table 1 - Examples from the SICK dataset

#	Example	Rel.	Ent.
1	A group of kids is playing in a yard and an old man is standing in the background A group of boys in a yard is playing and a man is standing in the background	4.5	Neutral
2	A man with a jersey is dunking the ball at a basketball game The ball is being dunked by a man with a jersey at a basketball game	4.9	Entailment
3	There is no biker jumping in the air A lone biker is jumping in the air	4.2	Contradiction

Table 1 includes a couple of examples extracted from the SICK 2014 dataset. Column *"Rel."* stands for relatedness and is a numeric value from 0 (totally unrelated) to 5 (extremely related), while column *"Ent."* stands for entailment and takes discrete values (N) - neutral, (C) - contradiction and (E) - entailment. It is obvious that detecting contradictions or entailment between two sentences is a fine-grained task, which requires

the model focusing on small details (such as simple negations) that would otherwise change the outcome of the prediction.

Given that the training objective of tripod is purely unsupervised, we designed two experiments to assess the usefulness of tripod vectors:

(A) Simple Multi-layer Perceptron (MLP): In this experiment we train a MLP to predict the entailment state between two sentences by feeding it their concatenated tripod vectors. Doing so allows us to see how well-suited the tripod captured context is in this granular task.

(B) LSTM encoders with auxiliary tripod vectors: In this experiment the measure the value of tripod vectors combined with fine-tuned task-specific models. Particularly, we rely on Gated Recurrent Units (GRUs) to walk over tokens in both sentences and we combine their final states through concatenation. We show results both with and without using tripod vectors for conditioning the final Softmax output. We also experiment with the GRU unit's input (see below).

For clarity (also see Figure 3), the list of our models is:

- **Model A:** MLP with tripod vectors;
- **Model B:** Parallel GRUs over locally computed embeddings for input tokens;
- **Model C:** Parallel GRUs over locally computed embeddings for input tokens and additional MLP with tripod input;
- **Model D:** Parallel GRUs over locally computed embeddings for input tokens and tripod vectors conditioning (by concatenating the embeddings with repeated tripod sentence vectors).

Other complex architectures achieve better results on this task - see (Radford et al., 2018; Kim et al.,2019; Zhang et al., 2018; Liu et al., 2019; Tan et al.,2018) and many others for details. However, our goal is to set our own baseline when doing the evaluation. Arguably, we could fine-tune our task specific models to yield better performance. However, this would be out-of-scope, since we are measuring the contribution of tripod vectors to this task, instead of going against the SOTA. The fact that we were able to directly use tripod vectors with the MLP for textual entailment, given the unsupervised training goal used in pre-training, is itself promising.

3.2. The Stanford Natural Language Inference (SNLI) corpus

SNLI (Bowman et al., 2015) is a well-known corpus for Natural Language Inference (NLI), which was extensively used to assess the performance of SOTA systems. Needless to say, it was also a good candidate for us, and we decided to further evaluate tripod and show the results obtained on this particular task and dataset.

The SNLI corpus contains 570K sentence pairs, which makes it 57 times larger than the SICK dataset. It does not include a relatedness score, but it includes fine-grained annotations at chunk level.

For simplicity, we used the exact same models described in the previous section (3.1) and we did not alter the train-validation-test split.

Table 2 shows the results obtained by our four models on the SICK and SNLI datasets. For completeness we include relevant SOTA results from comparable models.

Table 2 - Experimental results[4] for SICK and SNLI test datasets (accuracy on entailment prediction). "Sum of words" is from Bowman et al., (2015)

	Model	**SICK**	**SNLI**
Ours	**Model A**	73.68	77.83
	Model B	71.16	78.41
	Model C	72.20	72.51
	Model D	71.58	68.94
Other	**100d sum of words**	N/A	75.3
	Arora et al. (2016)	84.60	78.20

4. Conclusions and future work

We introduced a new model for extracting latent representation from sequences, that relies on a tripartite representation: summary, style and memory.

To evaluate the relevance of our model's learned representations we ran a series of experiments on several main-stream NLP tasks. We showed that the unsupervised training objective is sufficient to make the model usable on tasks that rely on fine-grained observations.

We argue that our model is smaller than several alternative models, it is computationally fast (we achieve 1K sequences per second on a RTX 2070) and the code is fully available as Opensource[5] under the Apache v2 License.

We also provide a python API and quick installation method via pip/conda packages, as well as collaborative notebooks (check the documentation uploaded on GitHub).

Work was done to evaluate Tripod vectors on JavaScript (JS) (latest model already available) and experiments are being conducted on detecting malicious Javascript code based on just latent representations.

Furthermore, we plan to extend our support to other languages and provide Wikipedia-based pre-trained models.

You can leave any suggestions and requests for support on the official GitHub repository.

[4] These tests are reported on an older Tripod model, not the one available on Github.

[5] https://github.com/adobe/tripod

References

lon, U., Zilberstein, M., Levy, O. and Yahav, E. (2019). code2vec: Learning distributed representations of code. Proceedings of the ACM on Programming Languages, 3(POPL):40.

Arora, S., Liang, Y. and Ma, T. (2016). A simple but tough-to-beat baseline for sentence embeddings.

Bowman, S. R., Angeli, G., Potts, C. and Manning, C. D. (2015). A large annotated corpus for learning natural language inference. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP). Association for Computational Linguistics.

Devlin, J., Chang, M.-W., Lee, K. and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional trans-formers for language understanding. arXiv preprintarXiv:1810.04805.

Gupta, M., Varma, V., et al. (2016). Doc2sent2vec: A novel two-phase approach for learning document representation. In Proceedings of the 39th International ACMSIGIR conference on Research and Development in Information Retrieval, pages 809–812. ACM.

Kim, S., Kang, I. and Kwak, N. (2019). Semantic sentence matching with densely-connected recurrent and co-attentive information. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 33,pages 6586–6593.

Le, Q. and Mikolov, T. (2014). Distributed representations of sentences and documents. In International conference on machine learning, pages 1188–1196.

Liu, X., He, P., Chen, W. and Gao, J. (2019). Multi-task deep neural networks for natural language understanding. arXiv preprint arXiv:1901.11504.

Marelli, M., Bentivogli, L., Baroni, M., Bernardi, R., Menini, S. and Zamparelli, R. (2014). Semeval-2014task 1: Evaluation of compositional distributional se-mantic models on full sentences through semantic relatedness and textual entailment. In Proceedings of the 8th international workshop on semantic evaluation (SemEval 2014), pages 1–8.

Merity, S., Xiong, C., Bradbury, J. and Socher, R. (2016). Pointer sentinel mixture models. arXiv preprintarXiv:1609.07843.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S. and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In Advances in neural information processing systems, pages 3111–3119

Pennington, J., Socher, R. and Manning, C. (2014). Glove: Global vectors for word representation. In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pages 1532–1543.

Peters, M. E., Neumann, M., Zettlemoyer, L. and Yih,W. (2018). Dissecting contextual word embeddings: Architecture and representation. arXiv preprintarXiv:1808.08949.

Radford,A., Narasimhan,K., Salimans, T. and Sutskever, I. (2018).Improving language understanding by generative pre-training.

Tan, C., Wei, F., Wang, W., Lv, W. and Zhou, M. (2018). Multiway attention networks for modeling sentence pairs. In IJCAI, pages 4411–4417.

Wang, Y., Stanton, D., Zhang, Y., Skerry-Ryan, R., Battenberg, E., Shor, J., Xiao, Y., Ren, F., Jia, Y. and Saurous, R. A. (2018). Style tokens: Unsupervised style modeling, control and transfer in end-to-end speech synthesis. arXiv preprint arXiv:1803.09017.

Zhang, Z., Wu, Y., Li, Z., He, S., Zhao, H., Zhou, X. and Zhou, X. (2018). I know what you want: Semantic learning for text comprehension. arXiv preprintarXiv:1809.02794.