Embedding User-Generated Content using Structural Supervision and Generative Models

Vinay Shukla[1,2] **Yang Yang**[1] **Siddarth Malreddy**[1] **Jinoo Baek**[1] **Dale Johnson**[1]
Wenfei Zhou[1] **Minh Pham**[1] **Mark Williams**[1] **Karthik Lakashman**[1]
[1]Google [2]University of California, Los Angeles
`vshukla@g.ucla.edu`
`{lizyang, malreddysid, jinoo, dalejohnson, wenfei}@google.com`
`{minphtx, wimark, lakshmanan}@google.com`

Abstract

One well-studied solution to avoid the need for vast amount of human-labeled data is to use self-supervised training objectives during pre-training, which enables learning on completely unlabeled examples. Especially in the case of larger models such as LLMs, these pre-training procedures have demonstrated benefits [Devlin et al., 2018]. In this work we focus on training LLMs for producing semantically expressive sentence embeddings for *User-Generated Content (UGC)* in comment-style mediums. We provide a novel self-supervised training paradigm that leverages the structure of comment data and also demonstrate the efficacy of LLM generation for producing quality training data. Through empirical evaluation, we show improvements against existing baselines methods on several downstream tasks.

1 Introduction

User-Generated Content (UGC) is a rich source of information on the Internet, with many forums and comment boards providing the most in-depth knowledge on niche topics. Building semantically expressive *Sentence Embeddings* for UGC has a huge potential to improve downstream tasks on such content like *classification*, *translation*, *question answering*, *summarization*, and *sentiment analysis*.

Comments-style UGC data has a very diverse linguistic structure and is often influenced by the context, language, and medium in which it is presented. For example, they naturally mix multiple author styles in multi-turn conversations, often intersperse multiple languages (like Hindi written in English), and contain contextual use of coded language and emojis. In this work, we focus on improving sentence embeddings for such UGC found in comment-style mediums, e.g. Youtube, Twitter, and Reddit.

Most relevant sentence-embedding approaches have relied on supervised-learning with human annotation to train models [Cer et al., 2018, Ni et al., 2021]. This procedure is both expensive and inefficient. We propose an *efficient* and *novel* method to train a “comment embedding model”. We summarize our contributions as follows: (i) we show that *using Large Language Models (LLMs)* to generate similar comment pairs for use in contrastive loss proves to work better than traditional methods trained on NLI (ii) show that *self-supervised learning tasks* that are *latent in “comment-style” UGC datasets* help to improve model performance.

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

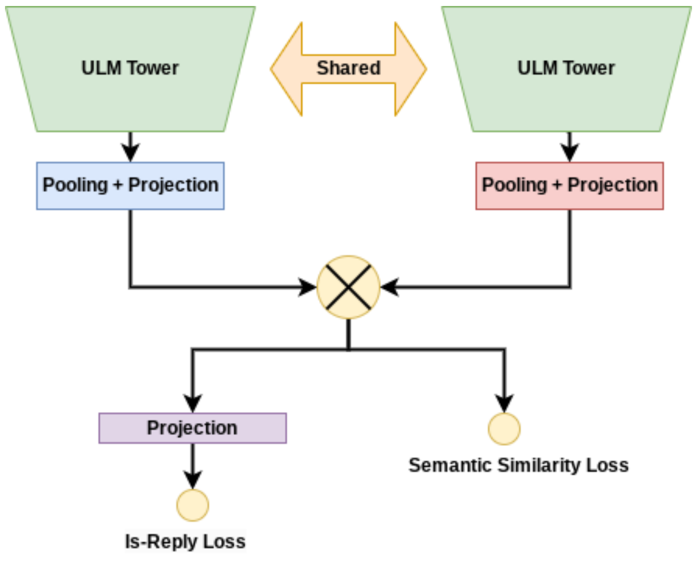

Figure 1: We present the general training structure for our pairwise learning. At inference time, the only piece that we have is a single ULM tower along with the immediate pooling and projection.

2 Related Work

Pretraining Objectives. For Large Langugage Models (LLM) there exist many popular unsupervised learning tasks that have been previously employed. The "masked language model" pretraining objective was proposed by Devlin et al. [2018] and involves masking out a certain percentage of words and allowing the model to try and reconstruct the phrase. Also proposed in Devlin et al. [2018] is the next-sentence prediction task, which involves predicting whether two sentences follow each other. Other tasks include next token prediction [Radford et al., 2018], contrastive learning [Chen et al., 2020] etc. The overarching idea is that labeling is expensive, which means that we want to find supervision among large pretraining datasets to help models achieve better semantic understanding.

Sentence Embeddings. The literature on sentence embeddings is quite extensive. Skip-Thought utilized surrounding sentence reconstruction as a task and an encoder-decoder model [Kiros et al., 2015]. This was eventually made quicker with Quick-Thought [Logeswaran and Lee, 2018]. Some effective methods simply combine word embeddings, which is seen in Arora et al. [2017]. Others siamese-network style approaches include InferSent [Conneau et al., 2017], SBERT [Reimers and Gurevych, 2019], SimCSE [Gao et al., 2021], and the siamese DAN + transformer models in [Yang et al., 2018]. More relevant to us are the two state-of-the-art approaches: Universal Sentence Encoder [Cer et al., 2018] and T5-Based Sentence Embeddings [Ni et al., 2021]. We note that both these approaches along with previous methods leverage the SNLI dataset [Bowman et al., 2015].

3 Method

3.1 Dual-Model Setup

Similar to previous siamese-network based approaches [Reimers and Gurevych, 2019, Conneau et al., 2017], we train with "two-models" that shares weights with one another. In our case, we choose a model based on PaLM 2 [Anil et al., 2023] which contains 1B parameters that we call ULM.

We define our training set $D = \{(x_i, y_i)\}$, where each sample $x_i = (x_i^1, x_i^2)$. Our goal is to train our model in order to successfully capture relationships between our text pairs (x_i^1, x_i^2). This is expressed through our objective functions or the tasks we are trying to learn at training time, which is shown in Figure 1 and detailed in later sections.

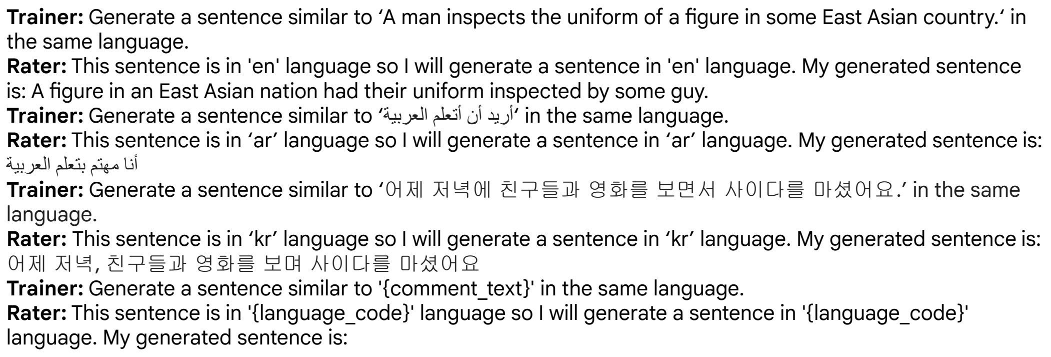
Trainer: Generate a sentence similar to 'A man inspects the uniform of a figure in some East Asian country.' in the same language.
Rater: This sentence is in 'en' language so I will generate a sentence in 'en' language. My generated sentence is: A figure in an East Asian nation had their uniform inspected by some guy.
Trainer: Generate a sentence similar to 'أريد أن أتعلم العربية' in the same language.
Rater: This sentence is in 'ar' language so I will generate a sentence in 'ar' language. My generated sentence is: أنا مهتم بتعلم العربية
Trainer: Generate a sentence similar to '어제 저녁에 친구들과 영화를 보면서 사이다를 마셨어요.' in the same language.
Rater: This sentence is in 'kr' language so I will generate a sentence in 'kr' language. My generated sentence is: 어제 저녁, 친구들과 영화를 보며 사이다를 마셨어요
Trainer: Generate a sentence similar to '{comment_text}' in the same language.
Rater: This sentence is in '{language_code}' language so I will generate a sentence in '{language_code}' language. My generated sentence is:

Figure 2: We use few-shot prompting with multiple languages as examples to make sure our data is multi-lingual. We note that comment_text is substituted for the new comment we are trying to evaluate and language_code is the langugage of the original comment.

3.2 Structured Signals

We define another task that relies on the innate structure of our "comment"-style UGC datasets. For pairs of comments, we can determine whether they are replies of each other. We note that this meta-information is easily tenable when doing a crawl over comment platforms.

Our model returns us a probability at the "is-reply" head which is shown in Figure 1. For a sample i, let $\hat{y}_i \in [0,1]$ correspond to the predicted probabilities of two comments being replies of each other and $y_i \in \{0,1\}$ be the desired label. For a batch $\mathcal{B}$, we take all samples that are part of the "is-reply" task and define this as $\mathcal{B}_{\text{reply}}$. As shown below, our loss becomes:

$$\mathcal{L}_{\text{reply}} = -\frac{1}{|\mathcal{B}_{\text{reply}}|} \sum_{i \in \mathcal{B}_{\text{reply}}} y_i \log \hat{y}_i + (1-y_i) \log(1-\hat{y}_i) \tag{1}$$

3.3 LLM Generation Supervision

We provide a second avenue of supervision that takes the form of LLM generated comment pairs. We prompt a LLM to give us a semantically similar comment to a base comment as seen in Figure 2. Hence, the training data for our task is composed of *semantically similar* pairs with no label.

We define a batch of training samples with labels $\mathcal{B} = \{(x_i, y_i)\}$ for $1 \le i \le k$, where k is the batch size. We take an *intermediate representation*, which is defined as a vector of length $n = 512$ that is obtained after passing passing x_i^1, the original comment, and x_i^2, the LLM generated comment, through the ULM model and performing the "pooling + projection" operations, which is shown in Figure 1. We define this representation as (v_i^1, v_i^2) for every $i \in \mathcal{B}$. Furthermore, we define a new batch $\mathcal{B}_{\text{sim}}$, consisting of only text pairs that correspond to this task.

We then create a distribution over our batch for each $i \in \mathcal{B}_{\text{sim}}$, i.e. we use an in-batch sampled softmax. We use cosine similarity as our scoring function and τ as a temperature constant. The main idea behind our loss is that we want the model to maximize $\text{sim}(v_i^1, v_i^2)$ while minimizing $\text{sim}(v_i^1, v_j^2)$ for i not equal to j.

$$\mathcal{L}_{\text{sim}} = \frac{1}{|\mathcal{B}_{\text{sim}}|} \sum_{i \in \mathcal{B}_{\text{sim}}} -\log\left(\frac{e^{\text{sim}(v_i^1, v_i^2)/\tau}}{\sum_{j \in \mathcal{B}_{\text{sim}}} e^{\text{sim}(v_i^1, v_j^2)/\tau}}\right) \tag{2}$$

4 Evaluation

In order to evaluate the model, we prepared several datasets to test the semantic representation capability of the embeddings. We treat the multi-lingual Universal Sentence Encoder [Cer et al., 2018] (USE-multi) as baseline. It should be noted that the USE-multi model supports only 16

Table 1: Model Evaluations for KNN recall on US and UK YT comments.

Model	YT Comments - US			YT Comments - UK		
	R@1	R@3	R@5	R@1	R@3	R@5
USE-multi	4.07%	7.56%	9.18%	4.47%	8.29%	10.16%
Ours	**4.63%**	**8.62%**	**10.44%**	**5.33%**	**9.67%**	**15.07%**

Table 2: Results for binary classification on the Reddit comment dataset for the politics vs. funny subreddits and aww vs. gaming subreddits. We report mean accuracy, AUC, and AUC-PR.

Method	Task	Model	Accuracy	AUC	AUC-PR
USE-multi	politics vs. funny	Logistic Regression	0.9040	0.9040	0.9235
Ours	politics vs. funny	Logistic Regression	**0.9053**	**0.9054**	**0.9257**
USE-multi	aww vs. gaming	Logistic Regression	0.8212	0.8215	0.8686
Ours	aww vs. gaming	Logistic Regression	**0.8240**	**0.8244**	**0.8697**

languages while our model is able to handle all major languages used in YT comments. We only use USE-multi as a baseline because its embedding dimension is equivalent to ours (512), so this will give us a meaningful comparison.

4.1 YouTube Comment Similarity Task

The YouTube comments dataset from Kaggle [J, 2017] has a list of comments sampled from various videos. It contains comments for videos from the US and the UK regions. In order to get semantically similar pairs we follow the same technique mentioned in Section 3.3 to synthetically generate text pairs. Since this process is very time consuming, we sample 10K comments from each region and generate the pairs. To evaluate the model, we run KNN in the generated set for every comment in the original dataset. Then we compute the recall@K which represents the percentage of comments in the original dataset that had their corresponding generated comment in the top K nearest neighbors.

Results and Discussion. Our results are tabulated in Table 1. We observe that on all recall@K values, our method outperforms the baseline. This is a testament to our method's clustering capability and how it is able to provide "close embeddings" for comments we expect to be similar in nature.

4.2 SubReddit Prediction Task

We derive an eval dataset from a publically available Reddit Kaggle comment dataset [Magnan, 2019]. In line with the tasks in the SentEval benchmark [Conneau and Kiela, 2018], we provide a series of downstream tasks that involves training a classifier on our model's embeddings. The goal of our task is binary classification, more specifically, we want to determine which subreddit a comment belongs to. In our tasks, we sample 4000 total comments from 2 random subreddits ensuring that there are exactly 2000 from each. We train a logistic regression classifier with 10-fold cross validation.

Results and Discussion. We report our evaluation in Table 2. Our method performs better than USE-multi on all metrics in both datasets for UGC content. Furthermore, USE-multi was pretrained on *reddit data* while ours was strictly trained on Youtube data, which makes this comparison inherently in the favor of USE-multi. Yet even with this in mind, our method achieves *higher downstream performance* on the fitted model.

5 Conclusion

In this paper, we have shown that the self-supervised training of Sentence Embeddings using LLM-generated datasets and structural signals latent in "comment datasets" has a potential to outperform existing baselines on UGC. As the pretraining utilizes a lot of data, using these techniques can help to greatly reduce the amount of *human labeled data*, which saves on time and cost. In the future, we hope to extend LLM-generated datasets to question-answering and NLI-based tasks. We refer the reader to Appendix A and B respectively for a more detailed discussion on future work and limitations of our approach.

References

Rohan Anil, Andrew M Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, et al. Palm 2 technical report. *arXiv preprint arXiv:2305.10403*, 2023.

Sanjeev Arora, Yingyu Liang, and Tengyu Ma. A simple but tough-to-beat baseline for sentence embeddings. In *International conference on learning representations*, 2017.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal, September 2015. Association for Computational Linguistics. doi: 10.18653/v1/D15-1075. URL `https://aclanthology.org/D15-1075`.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, et al. Universal sentence encoder. *arXiv preprint arXiv:1803.11175*, 2018.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

Alexis Conneau and Douwe Kiela. Senteval: An evaluation toolkit for universal sentence representations. *arXiv preprint arXiv:1803.05449*, 2018.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 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 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-1070. URL `https://aclanthology.org/D17-1070`.

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

Tianyu Gao, Xingcheng Yao, and Danqi Chen. Simcse: Simple contrastive learning of sentence embeddings. *arXiv preprint arXiv:2104.08821*, 2021.

Mitchell J. Trending youtube video statistics and comments. 2017. URL `https://www.kaggle.com/datasets/datasnaek/youtube`.

Ryan Kiros, Yukun Zhu, Russ R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Skip-thought vectors. *Advances in neural information processing systems*, 28, 2015.

Lajanugen Logeswaran and Honglak Lee. An efficient framework for learning sentence representations. *arXiv preprint arXiv:1803.02893*, 2018.

Sameul Magnan. 1 million reddit comments from 40 subreddits. 2019. URL `https://www.kaggle.com/datasets/smagnan/1-million-reddit-comments-from-40-subreddits`.

Jianmo Ni, Gustavo Hernández Ábrego, Noah Constant, Ji Ma, Keith B Hall, Daniel Cer, and Yinfei Yang. Sentence-t5: Scalable sentence encoders from pre-trained text-to-text models. *arXiv preprint arXiv:2108.08877*, 2021.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. 2018.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*, 2019.

Yinfei Yang, Steve Yuan, Daniel Cer, Sheng-yi Kong, Noah Constant, Petr Pilar, Heming Ge, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. Learning semantic textual similarity from conversations. In *Proceedings of the Third Workshop on Representation Learning for NLP*, pages 164–174, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-3022. URL `https://aclanthology.org/W18-3022`.

A Future Work

There are a lot of interesting follow up tasks to what we presented in this work. LLM generated supervision can be used on many other tasks other than just similar sentences. We plan to introduce tasks such as generating more comments in the style of a given commenter or using LLM to generate synthetic replies. Also, as introduced in the NLI dataset [Gao et al., 2021, Yang et al., 2018], we could prompt a LLM to generate "entailments" or "contradictory" sentences. Question-Answering tasks [Cer et al., 2018] have also been shown to improve Sentence Embeddings and is something that we would want to consider generating in the future.

Furthermore, we would like to alter the structure of our model structure. This involves different pooling operations instead of "last token" and also exploring different dimensionalities for our embedding projections.

B Limitations

A big limitation of our approach is the necessity of a LLM to prompt and generate our datasets. With the growth of LLMs in industry, many companies such as OpenAI provide APIs to directly access their models. Hence, we believe that although the resources needed to generate our synthetic dataset are certainly vital and require consideration, with the increasing accessability to LLMs and abundance of APIs[1] that allow for everyday users to access huge models, it is not inconceivable.

C Experimental Details

C.1 Hardware

All of our model training is done via Jax and trained on Cloud TPUs. The mesh shape is $[1, 64, 1]$, which means that we trained on 64 TPUs.

C.2 Hyperparameters

All our hyperparameters are chosen by using a 10% split on training data for validation. We use $\tau = \frac{1}{15}$ in our experiments and also a batch size of 512. We use 0.0002 as the constant learning rate, and Adafactor as the optimizer. Early stopping was performed by looking at batch avg cosine similarity in the evaluation dataset. We use Tanh as the activation function after the pooling and projection as shown in Figure 1. The pooling we do is "last token" as well, which makes sense because we use a decoder model. When we combine the output of the embeddings of the tower, we currently use concatenation and use a single projection to a head for the is-reply task. In each batch, we sample with equal probability from the "is-reply" task as well as the "semantic similarity task".

[1]https://openai.com/blog/openai-api