The title

Egorov Evgeny, Greenkina Tatyana, Potapova Polina

2021

1 Introduction

1. Explain validity of the problem 2. Explain key areas of improvements

2 Related Work

Classic embeddings like TF-IDF are poorly suited for the task of short text clustering, because they are too sparse. This problem can be solved by using low-dimensional embeddings - this is where autoencoders comes out. The article [4] proposes a method for training a neural network: low-dimensional SIF-embeddings are formed from the texts, then they are fed to the autoencoder, then the weights of the encoder and the clustering assignments are jointly adjusted. Clustering is done by applying k-means to the vectors produced by the encoder. Experiments show that adding an autoencoder helps to solve the clustering problem better than simply applying k-means to low-dimensional SIF-embeddings.

The authors of article [2] also investigate the problem of jointly clustering and learning representations. They suggest using the reconstruction loss of autoencoder to learn representations. And as a clustering loss, it is proposed to use the limit of the differentiable function, which allows one to train the model using SGD. The model, entitled Deep k-Means (DKM), that does not use a pre-training autoencoder is comparable to existing SoTA pretrained clustering models. The authors also showed that their pre-trained DKM model outperforms all similar clustering models, including the SoTA DCN model [10]. Authors shows their results in both image and text datasets.

Authors of this article [5] formulate a setting in which one could fine-tune a BERT model to unsupervised task of clustering texts. They tackle the problem by replacing one of the BERT self-supervised tasks by loss that induces clustering text embeddings around a set of vectors in embedding space. These vectors represent cluster centroids and the loss induces clustering of such vectors that are similar to that centroid vector. The authors motivate their choice of the loss function by allowing the model to adjust to domain-specific terms while learning a discriminate embedding space that separates all training set datapoints into well-separated areas.

Loss function in a task of clustering can consist of two parts: directly clustering loss and representation loss, which should prevent embeddings from overfitting. Authors of the article [1] suggest using the sum of contractive autoencoder loss and self-augmentation loss as a representation loss resulting in a new model DECCA. Self-augmentation loss minimizes the proximity between sample representation and its augmentation. Autoencoder loss is a sum of reconstruction Loss and regularizer that encourages the narrowing of the mapping to the feature space adjacent to the training data. Authors also propose to use KL divergence and the locality preserving loss as clustering loss. KL divergence helps cluster objects to be close to cluster centers. The locality preserving loss pushes adjacent data points together. In the end authors show that their approach over perform state-pf-the-art results on text and image datasets.

The article [9] works on clustering latent representations of images from MNIST, COIL datasets. Unlike previous solutions authors propose pairwise discriminative loss functions for latent representations.

This article [2] proposes to use autoencoder architecture for clustering of the data. Both image and text datasets participated in the study. Article is interesting for further realization as it allows to regularize the model during the training in the similar way as topic models.

A theoretical [6] article about deep connection between clustering loss and deep k-mean approaches. This article contains theoretical apparatus allowing to better understand and potentially modify the task of clustering texts and images.

This article [3] proposes a model (DEPICT) that is able to deal with noise in the data by introducing a uniform distribution that supposed to attract noisy data points. Sadly authors of this article tried their method only on image data.

2.1 Models

The following models were selected as representative to the perspective approach in text clusterization.

2.1.1 BERT baseline

Pre-trained language models have shown remarkable progress in many natural language understanding tasks. Especially, BERT applies the fine-tuning approach to achieve ground-breaking performance in a set of NLP tasks. BERT, a deep bidirectional transformer model, utilizes a huge unlabeled data to learn complex features and representations and then fine-tunes its pre-trained model on the downstream tasks with labeled data. For text classification we also use BERT model for a method to fine-tune pre-trained models unsupervisedly. A model simultaneously learns text representations and cluster assignments by jointly optimizing both the masked language model loss and the clustering oriented loss (Fig 1). The masked language model loss can help learn domain-specific knowledge and guarantee that our model not be misguided such as

all-zero vector. Clustering oriented loss is designed to make the latent representation space more separable.

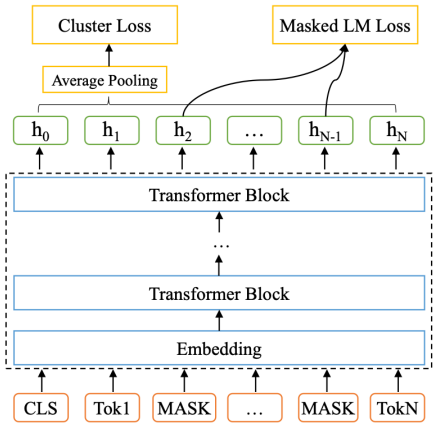

Figure 1: Unsupervised fine-tuning pre-trained model for text clustering

The loss function can be formulated as follows:

$$L = L_m + L_c \tag{1}$$

For this method we fine-tune our model as a masked-language model as in, which masks some of the input tokens randomly, and then predicts those masked tokens. The final hidden representations corresponding to the mask tokens are fed into a softmax layer over the vocabulary. The masked language model loss L_m is optimized by minimizing the negative log-likelihood.

The clustering loss with the representation z_i is designed to learn representation distribution with the help of an auxiliary target distribution. The clustering loss is defined as Kullback-Leibler (KL) divergence between distribution P and Q, where Q is the distribution of soft assignment by Student's t-distribution and P is the target distribution derived from Q. The clustering loss is defined as:

$$L_c = KL(P||Q) = \sum_i \sum_j p_{ij} log \frac{p_{ij}}{q_{ij}} \tag{2}$$

where q_{ij} is the similarity between text representation z_i and clustering centroid μ_j :

$$q_{ij} = \frac{(1 + ||z_i - \mu_j||^2/\alpha)^{-\frac{\alpha+1}{2}}}{\sum_{j'}(1 + ||z_i - \mu_{j'}||^2/\alpha)^{-\frac{\alpha+1}{2}}} \tag{3}$$

The target distribution p_{ij} puts more emphasis on data points assigned with high confidence and normalizes loss contribution. It is computed as follows:

$$p_{ij} = \frac{q_{ij}^2/f_j}{\sum_{j'} q_{ij'}^2/f_{j'}} \tag{4}$$

where $f_j = \sum_j q_{ij}$ are soft cluster frequencies. For clustering part, the target distribution P is derived from Q, so minimizing clustering loss is a self-training process. To initialize the cluster centroids, we first extract representations z_i through the original pre-trained model. Then employ standard k-means clustering in the representations space $\{z_i\}_{i=1}^n$ to obtain k initial centroids $\{z_j\}_{j=1}^k$. To avoid instability, we update the target distribution P, which depends on the predicted soft labels, per epoch rather than per batch. To make the target distribution P towards "groundtruth" distribution, we update P without masking any token.

2.1.2 Contrastive BERT

Unsupervised clustering aims at discovering the semantic categories of data according to some distance measured in the representation space. However, different categories often overlap with each other in the representation space at the beginning of the learning process, which poses a significant challenge for distance-based clustering in achieving good separation between different categories. To this end, this method propose Supporting Clustering with Contrastive Learning (SCCL) – a framework to leverage contrastive learning to promote better separation. As illustrated in Fig 2 the model consists of three components. A neural network $\psi(\cdot)$ first maps the input data to the representation space, which is then followed by two different heads $g()$ and $f()$ where the contrastive loss and the clustering loss are applied, respectively.

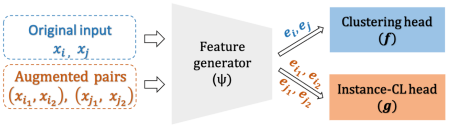

Figure 2: Training framework SCCL

A data consists of both the original and the augmented data. Specifically, for a randomly sampled minibatch $\mathcal{B} = \{x_i\}_{i=1}^M$, we randomly generate a pair of augmentations for each data instance in $\mathcal{B}$, yielding an augmented batch $\mathcal{B}^a$ with size $2M$, denoted as $\mathcal{B}^a = \{\tilde{x}_i\}_{i=1}^{2M}$.

For each minibatch $\mathcal{B}$, the Instance-CL loss is defined on the augmented pairs in $\mathcal{B}^a$. Let $i^1 \in \{1, \ldots, 2M\}$ denote the index of an arbitrary instance in augmented set $\mathcal{B}^a$, and let $i^2 \in \{1, \ldots, 2M\}$ be the index of the other instance in $\mathcal{B}^a$ augmented from the same instance in the original set $\mathcal{B}$. We refer to $\tilde{x}_{i^1}, \tilde{x}_{i^2} \in \mathcal{B}^a$ as a *positive* pair, while treating the other $2M$-2 examples in $\mathcal{B}^a$ as *negative* instances regarding this positive pair. Let $\tilde{z}_{i^1}$ and $\tilde{z}_{i^2}$ be the corresponding outputs of the head g, $\tilde{z}_j = g(\psi(\tilde{x}_j)), j = i^1, i^2$. Then for $\tilde{x}_{i^1}$, we try to separate $\tilde{x}_{i^2}$ apart from all negative instances in $\mathcal{B}^a$ by minimizing the following

$$\ell_{i^1}^I = -\log \frac{\exp(\text{sim}(\tilde{z}_{i^1}, \tilde{z}_{i^2})/\tau)}{\sum_{j=1}^{2M} \mathbb{1}_{j \neq i^1} \cdot \exp(\text{sim}(\tilde{z}_{i^1}, \tilde{z}_j)/\tau)}. \tag{5}$$

Here $\mathbb{1}_{j \neq i^1}$ is an indicator function and τ denotes the temperature parameter which we set as 0.5.

The Instance-CL loss is then averaged over all instances in $\mathcal{B}^a$,

$$\mathcal{L}_{\text{Instance-CL}} = \sum_{i=1}^{2M} \ell_i^I / 2M \ . \tag{6}$$

Loss of clustering is used similarly to the previous method. In summary, overall loss is

$$\mathcal{L} = \mathcal{L}_{\text{Instance-CL}} + \eta \mathcal{L}_{\text{Cluster}} = \sum_{j=1}^{M} \ell_j^C / M + \eta \sum_{i=1}^{2M} \ell_i^I / 2M.$$

2.1.3 AutoEncoders

Traditional clustering methods, *e.g.*, k-Means and Gaussian Mixture Models (GMMs), fully rely on the original data representations and may then be ineffective when the data points (*e.g.*, images and text documents) live in a high-dimensional space – a problem commonly known as the curse of dimensionality. Significant progress has been made in the last decade or so to learn better, low-dimensional data representations. The most successful techniques to achieve such high-quality representations rely on deep neural networks (DNNs), which apply successive non-linear transformations to the data in order to obtain increasingly high-level features. Auto-encoders (AEs) are a special instance of DNNs which are trained to embed the data into a (usually dense and low-dimensional) vector at the bottleneck of the network, and then attempt to reconstruct the input based on this vector. The appeal of AEs lies in the fact that they are able to learn representations in a fully unsupervised way. This approach (Deep k-Means) simultaneously allows you to train clustering on lower-dimensional data and maintain the accuracy of the reconstruction. In this approach, as in the previous methods considered, the loss of clustarization and reconstruction loss (autoencoder) are calculated together (Fig 3). The deep k-Means problem takes the following form:

$$\begin{aligned} &\min_{R,\theta} \sum_{x \in \mathcal{X}} g(x, A(x;\theta)) + \lambda f(\mathbf{h}_\theta(x), c_f(\mathbf{h}_\theta(x); \mathcal{R})), \\ &\text{with: } c_f(\mathbf{h}_\theta(x); \mathcal{R}) = \underset{r \in R}{argmin}\, f(\mathbf{h}_\theta(x), \mathbf{r}) \end{aligned} \tag{7}$$

g measures the error between an object x and its reconstruction $A(x;\theta)$ provided by an auto-encoder, θ representing the set of the auto-encoder's parameters. A regularization term on θ can be included in the definition of g. However, as most auto-encoders do not use regularization, we dispense with such a term here. $\mathbf{h}_\theta(x)$ denotes the representation of x in $\mathbb{R}^p$ output by the AE's encoder part

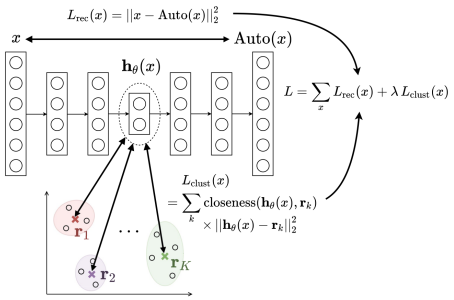

Figure 3: Deep k-Means approach instantiated with losses based on the Euclidean distance

and $f(\mathbf{h}_\theta(x), c_f(\mathbf{h}_\theta(x); \mathcal{R}))$ is the clustering loss corresponding to the k-Means objective function in the embedding space. Finally, λ in Problem (7) regulates the trade-off between seeking good representations for x *i.e.*, representations that are faithful to the original examples and representations that are useful for clustering purposes. Similar optimization problems can be formulated when f and g are similarity functions or a mix of similarity and distance functions. The approach proposed here directly applies to such cases.The following models were selected as representative to the perspective approach in text clusterization.

3 Experiments

3.1 Datasets

We used two datasets of short texts in our experiments. We split both of datasets in train, validation and test parts in ratio 50:25:25 correspondingly. We used text augmentation for experiments with contrastive model. We used an open-source Python library nlpaug [7] for that purpose. Double translation was chosen as the method of augmentation. In other words, we have translated the text from the original language into another language, and then back to the original one. We used model WMT'19 [8] to do so. We choose double translation method of augmentation because it showed the most correct results.

One of datasets is dataset of tweets in English that was described in [11]. We denote it as **Twitter**. The original dataset contained 89 categories and 2465 examples. We removed duplicates and classes consisting of only one element. After that the size of the dataset has reduced to 2353 elements and the number of classes became 82. To generate texts augmentations, we translated the original texts from English into German and back to English, as well as from English into Russian and back to English.

Interfax dataset contains news in Russian. We used titles of news for our experiments, not texts of news. The original dataset contains 48106 news and 6975 classes. We left only the 15 largest classes. Then the size of the dataset

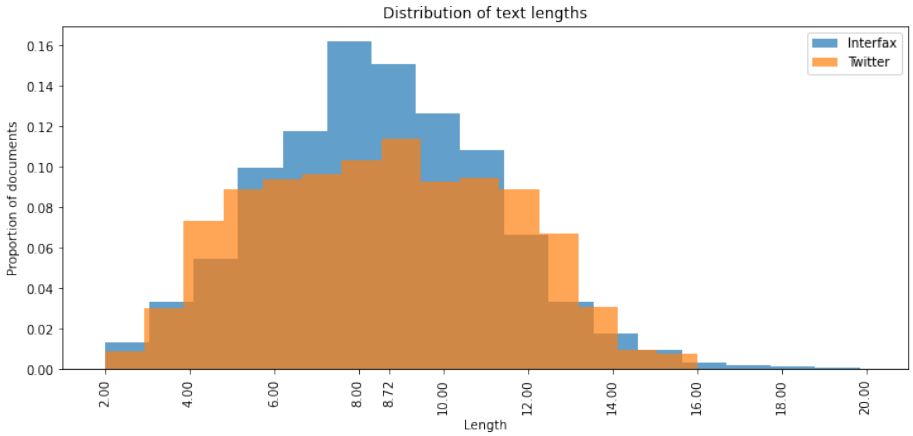

Figure 4: Distribution of text lengths

began to be 9883 examples. To augment texts we also used double translation from Russian to English and back.

One can found addition information about datasets in Table 3. Length distribution shows in 4.

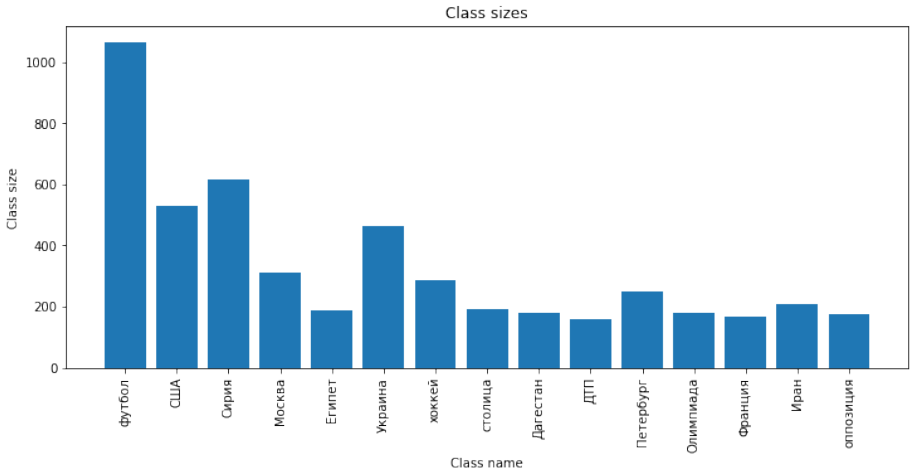

Figure 5: Class sizes

3.2 Metrics

Describe ARI and NMI.

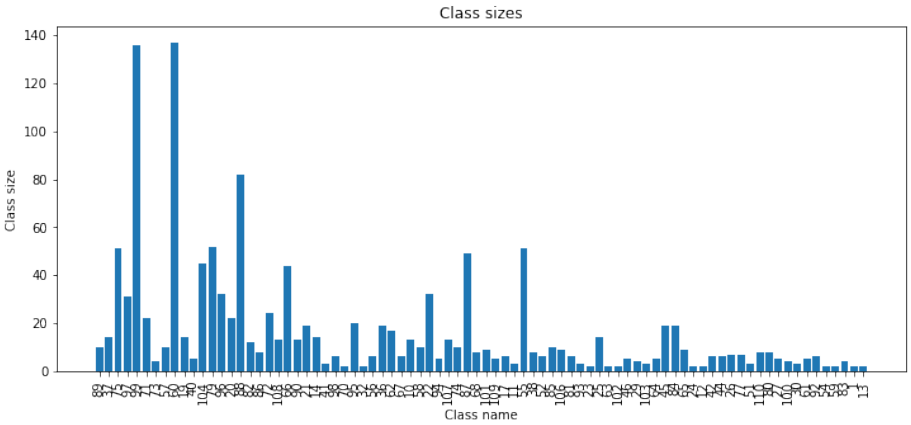

Figure 6: Class sizes

Table 1: Datasets description

	Twitter	Interfax
size of train part	1319	4966
size of val part	440	2446
size of test part	587	2471
size of vocabulary	11260	43305
average text size	8.54 ± 3.12	8.72 ± 2.58
min text size	2	2
max text size	16	23

3.3 Results

References

[1] Bassoma Diallo, Jie Hu, Tianrui Li, Ghufran Ahmad Khan, Xinyan Liang, and Yimiao Zhao. Deep embedding clustering based on contractive autoencoder. *Neurocomputing*, 433:96–107, 2021.

[2] Maziar Moradi Fard, Thibaut Thonet, and Eric Gaussier. Deep k-means: Jointly clustering with k-means and learning representations. *Pattern Recognition Letters*, 138:185–192, 2020.

[3] Kamran Ghasedi Dizaji, Amirhossein Herandi, Cheng Deng, Weidong Cai, and Heng Huang. Deep clustering via joint convolutional autoencoder embedding and relative entropy minimization. In *Proceedings of the IEEE international conference on computer vision*, pages 5736–5745, 2017.

Table 2: Metrics of models for Twitter and Interfax datasets

		Interfax		Twitter	
Losses types	**Embeddings**	**ARI**	**NMI**	**ARI**	**NMI**
Base Contrastive	BERT		0.447	0.513	0.763
Base Contrastive	RoBERTa		0.337		
Base Contrastive	T5			0.513	0.763
AutoEncoders Loss	AutoEncoders (TF-IDF)	0.193	0.362		
	AutoEncoders (BERT)			0.587	0.824

Table 3: Metrics of models for Twitter and Interfax datasets

		Interfax		Twitter	
Model	**Loss type**	**ARI**	**NMI**	**ARI**	**NMI**
BERT	Base Contrastive		0.447	0.513	0.763
RoBERTa	Base Contrastive		0.337		
T5	Base Contrastive			0.513	0.763
AutoEncoder (TF-IDF)	AutoEncoder	0.193	0.362		
AutoEncoder (BERT)				0.587	0.824

[4] Amir Hadifar, Lucas Sterckx, Thomas Demeester, and Chris Develder. A self-training approach for short text clustering. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pages 194–199, 2019.

[5] Shaohan Huang, Furu Wei, Lei Cui, Xingxing Zhang, and Ming Zhou. Unsupervised fine-tuning for text clustering. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5530–5534, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics.

[6] Mohammed Jabi, Marco Pedersoli, Amar Mitiche, and Ismail Ben Ayed. Deep clustering: On the link between discriminative models and k-means. *IEEE transactions on pattern analysis and machine intelligence*, 43(6):1887–1896, 2019.

[7] Edward Ma. Nlp augmentation. https://github.com/makcedward/nlpaug, 2019.

[8] Nathan Ng, Kyra Yee, Alexei Baevski, Myle Ott, Michael Auli, and Sergey Edunov. Facebook fair's wmt19 news translation task submission. In *Proc. of WMT*, 2020.

[9] Elad Tzoreff, Olga Kogan, and Yoni Choukroun. Deep discriminative latent space for clustering. *arXiv preprint arXiv:1805.10795*, 2018.

[10] Bo Yang, Xiao Fu, Nicholas D Sidiropoulos, and Mingyi Hong. Towards k-means-friendly spaces: Simultaneous deep learning and clustering. In *international conference on machine learning*, pages 3861–3870. PMLR, 2017.

[11] Jianhua Yin and Jianyong Wang. A dirichlet multinomial mixture model-based approach for short text clustering. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 233–242, 2014.