Proceedings of the 18th Conference on Natural Language Processing (KONVENS 2022)

DocSCAN: Unsupervised Text Classification via Learning from Neighbors

Dominik Stammbach
ETH Zurich
dominsta@ethz.ch

Elliott Ash
ETH Zurich
ashe@ethz.ch

Abstract

We introduce DocSCAN, a completely unsupervised text classification approach built on the *Semantic Clustering by Adopting Nearest-Neighbors* algorithm. For each document, we obtain semantically informative vectors from a large pre-trained language model. We find that similar documents have proximate vectors, so neighbors in the representation space tend to share topic labels. Our learnable clustering approach then uses pairs of neighboring datapoints as a weak learning signal to automatically learn topic assignments. On three different text classification benchmarks, we improve on various unsupervised baselines by a large margin.

1 Introduction

”What is this about?” is the starting question in human and machine reading of text documents. While this question would invite a variety of answers for documents in general, there is a large set of corpora for which each document can be labeled as belonging to a singular category or topic. Text classification is the task of automatically mapping texts into these categories. In the standard supervised setting (Vapnik, 2000), machine learning algorithms learn such a mapping from annotated examples. Annotating data is costly, however, and the resulting annotations are usually domain-specific. Unsupervised methods promise to reduce the number of labeled examples needed or to dispense with them altogether.

This paper builds on recent developments in the domain of unsupervised neighbor-based clustering of images, the SCAN algorithm: *Semantic Clustering by Adopting Nearest neighbors* (Van Gansbeke et al., 2020). We adapt the algorithm to text classification and report strong experimental results on three text classification benchmarks. The intuition behind SCAN is that images often share the same label, if their embeddings in some representation space are close to each other. Thus, we can leverage this regularity as a weakly supervised signal for training models. We encode a datapoint and its neighbors through a network where the output of the network is determined by a classification layer. The model learns that it should assign similar output probabilities to a datapoint and each of its neighbors. In the ideal case, model output is consistent and one-hot, i.e. the model confidently assigns the same label to two neighboring datapoints.

Deep Transformer networks have led to rapid improvements in text classification and other natural language processing (NLP) tasks (see e.g. Yang et al., 2019). We draw from such models to obtain task-agnostic contextualized language representations. We use SBERT embeddings (Reimers and Gurevych, 2019), which have proven performance in a variety of downstream tasks, such as retrieving semantically similar documents and text clustering. We show that in this semantic space, indeed neighboring documents tend to often share the same class label and we can use this proximity to build a dataset on which we apply our neighbor-based clustering objective. We find that training a model exploiting this regularity works well for text classification and outperforms a standard unsupervised baseline by a large margin. All code for DocSCAN can be found publicly available online.[1]

2 Related Work

Unsupervised learning methods are ubiquitous in natural language processing and text classification. For a more general overview, we refer to surveys discussing the topic in extensive details (see e.g. Feldman and Sanger, 2006; Grimmer and Stewart, 2013; Aggarwal and Zhai, 2012; Thangaraj

[1]https://github.com/dominiksinsaarland/DocSCAN

and Sivakami, 2018; Li et al., 2021). One common approach for text classification is to represent documents as vectors and then apply any clustering algorithm on the vectors (Aggarwal and Zhai, 2012; Allahyari et al., 2017). The resulting clusters can be interpreted as the text classification results. A popular choice is to use the k-means algorithm which learns cluster centroids that minimize the within-cluster sum of squared distances-to-centroids (see e.g. Jing et al., 2005; Guan et al., 2009; Balabantaray et al., 2015; Slamet et al., 2016; Song et al., 2016; Kwale, 2017). This methodology has also applications in social science research, where for example Demszky et al. (2019) classify tweets using this method. K-means can also be applied in an iterative manner (Rakib et al., 2020).

There exist more sophisticated methods for generating weak labels for unsupervised learning for text classification. However, most of these methods take into account some sort of domain knowledge or heuristically generated labels. For example, Ratner et al. (2017) generate a correlation-based aggregate of different labeling functions to generate proxy labels. Yu et al. (2020) create weak labels via heuristics, and Meng et al. (2020) use seed words (most importantly the label name) and infer the text category assignment from a masked language modeling task and seed word overlap for each category. DocSCAN is not subject to any of these dependencies. Similarly to k-means, we only need the number of topics present in a dataset. Hence, we think it is well suited to be compared against k-means.

3 Method

In this work, we build on the SCAN algorithm (Van Gansbeke et al., 2020). It is based on the intuition that a datapoint and its nearest neighbors in (some reasonable) representation space often share the same class label. The algorithm consists of three stages: (1) learn representations via a self-learning task, (2) mine nearest neighbors and fine-tune a network on the weak signal that two neighbors share the same label, and (3) confidence-based self-labeling of the training data (which is ommitted in this work[2]).

[2]The authors use heavily augmented images for the confidence-based self-labeling step. There is no straightforward translation of this approach to NLP. Tokens are discrete, symbolic characters, rather than the continuous quantities contained in pixels. We skip this step and leave exploration to future work.

Our adaptation DocSCAN to text classification works as follows. In Step 1, we need a document embedding method that serves as an analogue to SCAN's self-learning task for images. Textual Entailment (Dagan et al., 2005) is an interesting pretraining task yielding transferable knowledge and generic language representations, as already shown in (Conneau et al., 2018). Combining this pretraining task and large Transformer models, e.g., (Devlin et al., 2019) has led to SBERT (Reimers and Gurevych, 2019): A network of BERT models fine-tuned on the Stanford Natural Language Inference corpus (Bowman et al., 2015). SBERT yields embeddings for short documents with proven performance across domains and for a variety of tasks, such as semantic search and clustering. For a given corpus, we apply SBERT and get a 768-dimensional dense vector for each document.[3] We directly use the pre-trained SBERT model fine-tuned on top of the MPNet model[4] (Song et al., 2020), which yields the best[5] (on average) performing embeddings for 14 sentence embedding tasks and 6 semantic search tasks.

Step 2 is the mining of neighbors in the embedding space. We apply Faiss (Johnson et al., 2017) to get Euclidean distances between all embedded document vectors. The retrieved neighbors are the documents having the smallest Euclidean distance to a reference datapoint.

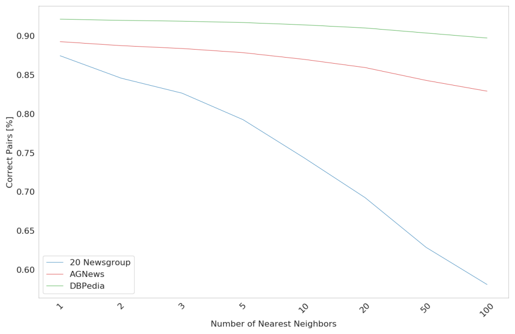

Figure 1: Accuracy of datapoint/neighbor pairs sharing the same label for different text classification benchmarks.

SCAN worked because images with proximate embeddings tended to share class labels. Is that

[3]We also experimented with other document representations. We discuss results in more detail in Appendix B

[4]The *all-mpnet-base-v2* model taken from `https://www.sbert.net/docs/pretrained_models.html`

[5]"best" embeddings at the time of submission of this work

the case with text? Figure 1 shows that the answer is yes: across three text classification benchmarks, neighboring document pairs do indeed often share the same label. The fraction of pairs sharing the same label at $k = 1$ is above 85% for all datasets examined. For $k = 5$, the resulting fraction of correct pairs (from all mined pairs) is still higher than 75% in all cases. Furthermore, these frequencies of correct pairs for $k = 5$ are often higher than the frequency of correct pairs reported for images in (Van Gansbeke et al., 2020).

Next, we describe the SCAN loss,

$$-\frac{1}{|\mathcal{D}|}\sum_{x\in\mathcal{D}}\sum_{k\in\mathcal{N}_x} log(f(x)\cdot f(k)) + \lambda\sum_{i\in\mathcal{C}} p_i log(p_i) \tag{1}$$

which can be broken down as follows. The first part of Eq. (1) is the consistency loss. Our model f (parametrized by a neural net) computes a label for a datapoint x from the dataset $\mathcal{D}$ and for each datapoint k in the set of the mined neighbors from x in $\mathcal{N}_x$. We then simply compute the dot product (denoted as $\cdot$) between the output distribution (normalized by a softmax function) for our datapoint x and its neighbor k. This dot product is maximized if both model outputs are one-hot with all probability mass on the same entry in the respective vectors. It is consistent because we want to assign the same label for a datapoint and all its neighbors. The second term is an auxiliary loss to obtain regularization via entropy (scaled by a weight λ), such that the model is encouraged to spread probability mass across all clusters $\mathcal{C}$ where p_i denotes the assigned probability of cluster i in $\mathcal{C}$ by the model. Without this entropy term, there exists a shortcut by collapsing all examples into one single cluster. The entropy term ensures that the distribution of class labels resulting from applying DocSCAN tends to be roughly uniform. Thus, it works best for text classification tasks where the number of examples per class is balanced as well.

To summarize: We use SBERT and embed every datapoint in a given text classification dataset. We then mine the five nearest neighbors for every datapoint. This yields our weakly supervised training set. We fine-tune networks on neighboring datapoints using the SCAN loss. At test time, we compute $f(x)$ for every datapoint x in the test set. We set the number of outcome classes equal to the numbers of classes in our considered datasets and use the hungarian matching algorithm (Kuhn and Yaw, 1955) to obtain the optimal cluster-to-label assignment.

4 Experiments

We apply DocSCAN on three widely used but diverse text classification benchmarks: The 20NewsGroup data (Lang, 1995), the AG's news corpus (Zhang et al., 2015), and lastly the DBPedia ontology dataset (Lehmann et al., 2015). We provide further dataset descriptions in Appendix Section A.

The main results are reported in Table 1. For all experiments, we report the mean accuracy over 10 runs on the test set (with different seeds and the 95% confidence interval). The columns correspond to the benchmark corpora. The rows correspond to the models, starting with a random baseline [1], two k-means baselines [2, 3] and the results obtained by DocSCAN in [4]. We also report a supervised learning baseline [5] and results taken from related literature in [6].

Row [1] provides a sensible lower-bound, row [5] analogously a supervised upper-bound for text classification performance. In the random draw [1], accuracy by construction converges to the average of the class proportions. The supervised model [5] is an SVM classifier applied to the same SBERT embeddings[6] which serve as inputs to the k-means baseline and to DocSCAN. Predictably, the supervised baseline obtains strong accuracy on these benchmark classification tasks.

The industry workhorse for clustering is k-means, an algorithm for learning cluster centroids that minimize the within-cluster sum of squared distances-to-centroids. When applied to TF-IDF-weighted bag-of-n-grams features [2], k-means improves over the results obtained in [1]. When applied to SBERT vectors [3], we see large improvements over all previous experiments. These results suggest that k-means applied to reasonable document embeddings already yields satisfactory results for text classification. Second, they corroborate what we already saw in Figure 1, that neighbors in SBERT representation space contain information about text topic classes.

So what does DocSCAN add? We fine-tune a classification layer using the SBERT embeddings with the SCAN objective (Eq. 1) and $k = 5$ neighbors. We observe unambiguous and significant improvements over the already strong k-means base-

[6]We also trained the SVM classifier with TF-IDF representations and obtained similar results for all experiments.

Experiment	20 News	AG news	DBPedia
[1] Random Baseline	7.0 ±0.0	26.1 ±0.3	7.7 ±0.0
[2] TF-IDF + k-means	32.6 ±1.1	49.5 ±6.3	47.6 ±3.0
[3] SBERT embeddings + k-means	54.2 ±1.6	69.2 ±7.3	76.9 ±4.3
[4] DocSCAN	**59.4** ±1.9	**84.1** ±2.6	**84.6** ±3.8
[5] SBERT embeddings + SVM	82.7	92.1	98.7
6] Related Literature	58.2	84.52 ±0.50	91.1

Table 1: Test-set accuracy by benchmark dataset (columns) and classifier (rows). Cell values give the mean over 10 runs with 95% confidence interval. Note that the results reported from the related literature in the last row might not be directly comparable to our method due to different experimental setups. The 20 News results are taken from (Chu et al., 2021), the AG news results from (Rakib et al., 2020), and the DBPedia results from (Meng et al., 2020).

line in all three datasets (as we can judge from the 95% confidence intervals). The smallest improvements (over 5% points) are made on the 20 News dataset, containing 20 classes. The largest improvement gains are observed for AG news with 4 classes, suggesting that DocSCAN above all works best for text classification tasks with a lower number of classes. Surprisingly, we do not find that the improvements correlate with the accuracy of neighboring pairs sharing the same label (see Figure 1), but rather with the numbers of classes in the dataset (see Table 2). In the case of the AG news data with only a few different classes, we find that DocSCAN approaches the performance of a supervised baseline using the same input features.

Finally, in [6] we show results from related literature on unsupervised text classification. We find that DocSCAN performs comparable to other completely unsupervised methods. We find that DocSCAN obtains the best results for the 20 News dataset, comparable results in the case of AG news data and slightly worse results than the related literature on the DBPedia data. However, we note that DocSCAN is a simple method consisting of only $hidden\_dim * num\_classes$ parameters, that is exactly one classification layer which is fine-tuned in a completely unsupervised manner using the SCAN loss. Whereas the results for DBPedia from (Meng et al., 2020) are obtained by fine-tuning whole language models using domain knowledge (seed words).

We show and discuss ablation experiments for DocSCAN in Appendix B. Specifically, we conduct experiments regarding the various hyperparameters of the algorithm and find that it is robust to such choices. Furthermore, we find that DocSCAN outperforms a k-means baseline over different input features in all settings. Given the findings derived from these experiments, we recommend default hyperparameters for applying DocSCAN.

5 Conclusion

In this work, we introduced DocSCAN for unsupervised text classification. Analogous to the recognizable object content of images, we find that a document and its close neighbors in embedding space often share the same class in terms of the topical content. We show that this consistency can be used as a weak signal for fine-tuning text classifier models in an unsupervised fashion. We start with SBERT embeddings and fine-tune DocSCAN on three text classification benchmarks. We outperform a random baseline and two k-means baselines by a large margin. We discuss the influence of hyper-parameters and input features for DocSCAN and recommend default parameters which we have observed to work well across our main results.

As with images, unsupervised learning with SCAN can be used for text classification. However, the method may not work as generically, and should for example be limited to text classification in cases of balanced datasets (given that we use an entropy loss as an auxiliary objective). Still, this work points to the promise of further exploration of unsupervised methods using embedding geometry.

References

Charu C. Aggarwal and ChengXiang Zhai. 2012. *A Survey of Text Clustering Algorithms*, pages 77–128. Springer US, Boston, MA.

Mehdi Allahyari, Seyedamin Pouriyeh, Mehdi Assefi, Saied Safaei, Elizabeth D. Trippe, Juan B. Gutierrez, and Krys Kochut. 2017. A brief survey of text mining: Classification, clustering and extraction techniques.

Ricardo A. Baeza-Yates and Berthier A. Ribeiro-Neto. 1999. *Modern Information Retrieval*. ACM Press / Addison-Wesley.

Rakesh Chandra Balabantaray, Chandrali Sarma, and Monica Jha. 2015. Document clustering using k-means and k-medoids.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 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.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Brian Strope, and Ray Kurzweil. 2018. Universal sentence encoder for English. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 169–174, Brussels, Belgium. Association for Computational Linguistics.

Zewei Chu, Karl Stratos, and Kevin Gimpel. 2021. Unsupervised label refinement improves dataless text classification. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4165–4178, Online. Association for Computational Linguistics.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loic Barrault, and Antoine Bordes. 2018. Supervised learning of universal sentence representations from natural language inference data.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The pascal recognising textual entailment challenge. In *Proceedings of the First International Conference on Machine Learning Challenges: Evaluating Predictive Uncertainty Visual Object Classification, and Recognizing Textual Entailment*, MLCW'05, page 177–190, Berlin, Heidelberg. Springer-Verlag.

Dorottya Demszky, Nikhil Garg, Rob Voigt, James Zou, Matthew Gentzkow, Jesse Shapiro, and Dan Jurafsky. 2019. Analyzing polarization in social media: Method and application to tweets on 21 mass shootings.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Ronen Feldman and James Sanger. 2006. *Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data*. Cambridge University Press, USA.

Justin Grimmer and Brandon M Stewart. 2013. Text as data: The promise and pitfalls of automatic content analysis methods for political texts. *Political Analysis*, page mps028.

Hu Guan, Jingyu Zhou, and Minyi Guo. 2009. A class-feature-centroid classifier for text categorization. In *Proceedings of the 18th International Conference on World Wide Web*, WWW '09, page 201–210, New York, NY, USA. Association for Computing Machinery.

Liping Jing, Michael K. Ng, Jun Xu, and Joshua Zhexue Huang. 2005. Subspace clustering of text documents with feature weighting k-means algorithm. In *Advances in Knowledge Discovery and Data Mining*, pages 802–812, Berlin, Heidelberg. Springer Berlin Heidelberg.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2017. Billion-scale similarity search with gpus.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. Cite arxiv:1412.6980Comment: Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015.

H. W. Kuhn and Bryn Yaw. 1955. The hungarian method for the assignment problem. *Naval Res. Logist. Quart*, pages 83–97.

Francis Musembi Kwale. 2017. A critical review of k means text clustering algorithms. *International Journal of Advanced Research in Computer Science*, 4(9):27–34.

Ken Lang. 1995. Newsweeder: Learning to filter netnews. In *Proceedings of the Twelfth International Conference on Machine Learning*, pages 331–339.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, and Christian Bizer. 2015. DBpedia - a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web Journal*, 6(2):167–195.

Qian Li, Hao Peng, Jianxin Li, Congying Xia, Renyu Yang, Lichao Sun, Philip S. Yu, and Lifang He. 2021. A survey on text classification: From shallow to deep learning.

Yu Meng, Yunyi Zhang, Jiaxin Huang, Chenyan Xiong, Heng Ji, Chao Zhang, and Jiawei Han. 2020. Text classification using label names only: A language model self-training approach. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9006–9017, Online. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Md Rashadul Hasan Rakib, Norbert Zeh, Magdalena Jankowska, and Evangelos Milios. 2020. Enhancement of short text clustering by iterative classification. In *Natural Language Processing and Information Systems*, pages 105–117, Cham. Springer International Publishing.

Alexander Ratner, Stephen H. Bach, Henry Ehrenberg, Jason Fries, Sen Wu, and Christopher Ré. 2017. Snorkel. *Proceedings of the VLDB Endowment*, 11(3):269–282.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks.

Cepy Slamet, Ali Rahman, Muhammad Ali Ramdhani, and Wahyudin Darmalaksana. 2016. Clustering the verses of the holy qur'an using k-means algorithm.

Jia Song, Xianglin Huang, Sijun Qin, and Qing Song. 2016. A bi-directional sampling based on k-means method for imbalance text classification. In *2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS)*, pages 1–5.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. Mpnet: Masked and permuted pre-training for language understanding.

M. Thangaraj and M Sivakami. 2018. Text classification techniques: A literature review. *Interdisciplinary Journal of Information, Knowledge, and Management*, 13:117–135.

Wouter Van Gansbeke, Simon Vandenhende, Stamatios Georgoulis, Marc Proesmans, and Luc Van Gool. 2020. Scan: Learning to classify images without labels. In *Proceedings of the European Conference on Computer Vision*.

Vladimir Vapnik. 2000. *The Nature of Statistical Learning Theory*. Springer: New York.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems*, volume 32, pages 5753–5763. Curran Associates, Inc.

Yue Yu, Simiao Zuo, Haoming Jiang, Wendi Ren, Tuo Zhao, and Chao Zhang. 2020. Fine-tuning pre-trained language model with weak supervision: A contrastive-regularized self-training approach.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, volume 28, pages 649–657. Curran Associates, Inc.

A Dataset Statistics

Dataset	# Examples	# Classes	Avg. Length	Example
20News	11’314	20	248	[...] I Have a Sound Blaster ver 1.5 When I try to install driver ver 1.5 (driver that comes with window 3.1) [...]
AG’s Corpus	120’000	4	31	Wall St. Bears Claw Back Into the Black (Reuters) Reuters - Short-sellers, Wall Street’s dwindling band of ultra-cynics, are seeing green again.
DBPedia	560’000	14	46	Abbott of Farnham E D Abbott Limited was a British coachbuilding business based in Farnham Surrey trading under that name from 1929. A major part of their output was under sub-contract to motor vehicle manufacturers. Their business closed in 1972.

Table 2: Dataset Statistics

We apply DocSCAN to three diverse datasets widely used in unsupervised text classification: (1) The 20NewsGroup data contains text from UseNet discussion groups (20 classes). (2) The AG’s news corpus (Zhang et al., 2015), which consists of the title and description field of news articles (4 classes). And lastly the DBPedia ontology dataset (Lehmann et al., 2015) which includes titles and abstracts of Wikipedia articles (14 classes).

In Table 2, we show the numbers of training examples, number of classes, the average document length and one text example from each dataset. We selected these datasets because they are established standard datasets for unsupervised text classification. The three datasets vary in domain, number of classes, and text lengths. But they have all in common that the number of examples per class are roughly balanced, hence DocSCAN is well suited to tackle these datasets.

B Ablation Experiments

In Table 3, we report how DocSCAN performs under various different hyper-parameters which possibly could affect the performance of the algorithm. In the two last columns, we report the mean accuracy of 10 runs (and the 95% confidence interval) on the AG news and DBPedia training datasets. As common practice in unsupervised learning, we cluster the dataset and then report evaluation metrics on the training set itself (whereas in the main results, we discuss the performance on the test sets of the respective datasets).

We investigate the number of neighbors considered (A), the weight of the entropy loss (B), batch sizes (C), dropout (D) and number of epochs (E). To optimize the SCAN loss, we use Adam (Kingma and Ba, 2014) with default parameters in all experiments. DocSCAN runs somewhat stable across different choices of these hyperparameters, yielding similar results which all outperform the k-means baseline by a large margin. The two worst performances are achieved if we either set the entropy weight too low ($\lambda = 1$) or do not consider enough neighboring pairs ($k = 2$). The influence of all other hyperparameters seems limited. We recommend using the default parameters reported in the first row. The main results in Table 1 were obtained using this set of hyperparameters.

We also investigate whether the success of DocSCAN for text classification stems from the chosen document embeddings. For this, we consider a number of different input features for the algorithm and run DocSCAN with these features, holding everything else constant. We show results in Table 4. We report the mean performance of 10 runs and the 95% confidence interval on the AG news training set.

We run several document embedding techniques, starting with the TF-IDF-weighted bag-of-n-grams (Baeza-Yates and Ribeiro-Neto, 1999). Second, we consider the averaged GloVe embeddings of all words in a document (Pennington et al., 2014), Universal Sentence Encoder (USE) embeddings (Cer et al., 2018) and lastly the performance of DocSCAN using SBERT embeddings (Reimers and Gurevych, 2019). We observe again that DocSCAN performs better than k-means in every setting. However, the performance gap for different features varies. For example, we observe best k-means performance using USE embeddings, whereas the best DocSCAN performance is achieved via SBERT embeddings. Also, TF-IDF + k-means yields rather mixed results, whereas TF-IDF + DocSCAN performs more than 20%

	Neighbors	Entropy Weight	Batch Size	Dropout	Epochs	Accuracy AG news	Accuracy DBPedia
DocSCAN	5	2	128	0.1	5	83.2 ±3.8	85.8 ±3.5
(A)	2					77.5 ±6.7	83.1 ±5.1
	3					78.4 ±5.5	85.3 ±3.0
	10					82.4 ±5.6	86.1 ±3.5
(B)		1				75.8 ±5.3	80.3 ±2.8
		4				80.4 ±3.5	86.7 ±2.8
(C)			64			82.4 ±5.0	87.5 ±4.2
			256			81.3 ±4.3	84.6 ±4.1
(D)				0		81.9 ±3.7	86.1 ±4.4
				0.33		80.3 ±3.9	86.8 ±2.6
(E)					3	79.4 ±5.3	84.2 ±4.4
					10	81.5 ±3.4	84.7 ±3.7
k-means						66.2 ±8.2	77.1 ±4.9

Table 3: Ablation Studies for DocSCAN Hyper-parameters (results reported on the AG news and DBPedia training set, cell values give the mean over 10 runs with 95% confidence interval).

points better. In light of these results, we recommend to use SBERT embeddings if considering applying DocSCAN to other work.

Features	k-means	DocSCAN
TF-IDF	53.9 ±4.1	76.8 ±4.3
avg. GloVe	55.4 ±3.6	59.3 ±0.3
USE Embeddings	74.4 ±8.3	79.1 ±8.6
SBERT	66.2 ±8.2	83.2 ±3.8

Table 4: Ablation Studies for Different Input Features (results reported on the AG news training set, cell values give the mean over 10 runs with 95% confidence interval).