Event-Driven News Stream Clustering using Entity-Aware Contextual Embeddings

Kailash Karthik Saravanakumar[*2] **Miguel Ballesteros**[1]
Muthu Kumar Chandrasekaran[1] **Kathleen McKeown**[1,2]
[1]Amazon AI, USA
[2]Department of Computer Science, Columbia University, NY, USA
{ballemig, cmuthuk, mckeownk}@amazon.com
{kailashkarthik.s}@columbia.edu

Abstract

We propose a method for online news stream clustering that is a variant of the non-parametric streaming K-means algorithm. Our model uses a combination of sparse and dense document representations, aggregates document-cluster similarity along these multiple representations and makes the clustering decision using a neural classifier. The weighted document-cluster similarity model is learned using a novel adaptation of the triplet loss into a linear classification objective. We show that the use of a suitable fine-tuning objective and external knowledge in pre-trained transformer models yields significant improvements in the effectiveness of contextual embeddings for clustering. Our model achieves a new state-of-the-art on a standard stream clustering dataset of English documents.

1 Introduction

Human presentation and understanding of news articles is almost never isolated. Seminal real-world events spawn a chain of strongly correlated news articles that form a news story over time. Given the abundance of online news sources, the consumption of news in the context of the stories they belong to is challenging. Unless people are able to scour the many news sources multiple times a day, major events of interest can be missed as they occur. The real-time monitoring of news, segregating articles into their corresponding stories, thus enables people to follow news stories over time.

This goal of identifying and tracking topics from a news stream was first introduced in the Topic Detection and Tracking (TDT) task (Allan et al., 1998). Topics in the news stream setting usually correspond to real-world events, while news articles may also be categorized thematically into sports, politics, etc. We focus on the task of clustering news on the basis of event-based story chains. We make a distinction between our definition of an *event topic*, which follows TDT and refers to large-scale real-world events, and the fine-grained events used in trigger-based event detection (Ahn, 2006). Given the non-parametric nature of our task (the number of events is not known beforehand and evolves over time), the two primary approaches have been topic modeling using Hierarchical Dirichlet Processes (HDPs) (Teh et al., 2005; Beykikhoshk et al., 2018) and Stream Clustering (MacQueen, 1967; Laban and Hearst, 2017; Miranda et al., 2018). While HDPs use word distributions within documents to infer topics, stream clustering models use representation strategies to encode and cluster documents. Contemporary models have adopted stream clustering using TF-IDF weighted bag of words representations to achieve state-of-the-art results (Staykovski et al., 2019).

In this paper, we present a model for event topic detection and tracking from news streams that leverages a combination of dense and sparse document representations. Our dense representations are obtained from BERT models (Devlin et al., 2019) fine-tuned using the triplet network architecture (Hoffer and Ailon, 2015) on the event similarity task, which we describe in Section 3. We also use an adaptation of the triplet loss to learn a Support Vector Machine (SVM) (Boser et al., 1992) based document-cluster similarity model and handle the non-parametric cluster creation using a shallow neural network. We empirically show consistent improvement in clustering performance across many clustering metrics and significantly less cluster fragmentation.

The main contributions of this paper are:

- We present a novel technique for event-driven news stream clustering, which, to the best of our knowledge, is the first attempt of using

[*]Work done during internship at Amazon

contextual representations for this task.

- We investigate the impact of BERT's fine-tuning objective on clustering performance and show that tuning on the event similarity task using triplet loss improves the effectiveness of embeddings for clustering.
- We demonstrate the importance of adding external knowledge to contextual embeddings for clustering by introducing entity awareness to BERT. Contrary to a previous claim (Staykovski et al., 2019), we empirically show that dense embeddings improve clustering performance when augmented with task-pertinent fine-tuning, external knowledge and the conjunction of sparse and temporal representations.
- We analyze the problem of cluster fragmentation and show that it is not captured well by the metrics reported in the literature. We propose an additional metric that captures fragmentation better and report results on both.

2 Related Work

In this section, we introduce the TDT task, prior work on tracking events from news streams and a few related parametric variants of the TDT task.

The goal of the TDT task is to organize a collection of news articles into groups called topics. Topics are defined as sets of highly correlated news articles that are related to some seminal real-world event. This is a narrower definition than the general notion of a topic which could include subjects (like *New York City*) as well. TDT defines an event to be represented by a triple <location, time, people involved>, which spawns a series of news articles over time. We are interested in all five sub-tasks of TDT - story segmentation, first story detection, cluster detection, tracking and story link detection - though we do not explicitly tackle these sub-problems individually.

After the initial work on the TDT corpora, interest in news stream clustering was rekindled by the news tracking system *NewsLens* (Laban and Hearst, 2017). NewsLens tackled the problem in multiple stages: (1) document representation through keyphrase extraction; (2) non-parametric batch clustering using the Louvian algorithm (Blondel et al., 2008); and (3) linking of clusters across batches. Staykovski et al. (2019) presented a modified version of this model, using TF-IDF bag of words document representations instead of keywords. They also compared the relative performance of sparse TF-IDF bag of words and dense doc2vec (Le and Mikolov, 2014) representations and showed that the latter performs worse, both individually and in unison with sparse representations. Linger and Hajaiej (2020) extended this batch clustering idea to the multilingual setting by incorporating a Siamese Multilingual-DistilBERT (Sanh et al., 2019) model to link clusters across languages.

In contrast to the batch-clustering approach, Miranda et al. (2018) adopt an online clustering paradigm, where streaming documents are compared against existing clusters to find the best match or to create a new cluster. We adopt this stream clustering approach as it is robust to temporal density variations in the news stream. Batch clustering models tune a batch size hyperparameter that is both training corpus dependent and might not be able to adjust to temporal variations in stream density. In their model, they also use a pipeline architecture, having separate models for document-cluster similarity computation and cluster creation. Similarity between a document and cluster is computed along multiple document representations and then aggregated using a Rank-SVM model (Joachims, 2002). The decision to merge a document with a cluster or create a new cluster is taken by an SVM classifier. Our model also follows this architecture, but critically adds dense document representations, an SVM trained on the adapted triplet loss for aggregating document-cluster similarities and a shallow neural network for cluster creation.

News event tracking has also been framed as a non-parametric topic modeling problem (Zhou et al., 2015) and HDPs that share parameters across temporal batches have been used for this task (Beykikhoshk et al., 2018). Dense document representations have been shown to be useful in the parametric variant of our problem, with neural LDA (Dieng et al., 2019a; Keya et al., 2019; Dieng et al., 2019b; Bianchi et al., 2020), temporal topic evolution models (Zaheer et al., 2017; Gupta et al., 2018; Zaheer et al., 2019; Brochier et al., 2020) and embedding space clustering (Momeni et al., 2018; Sia et al., 2020) being some prominent approaches in the literature.

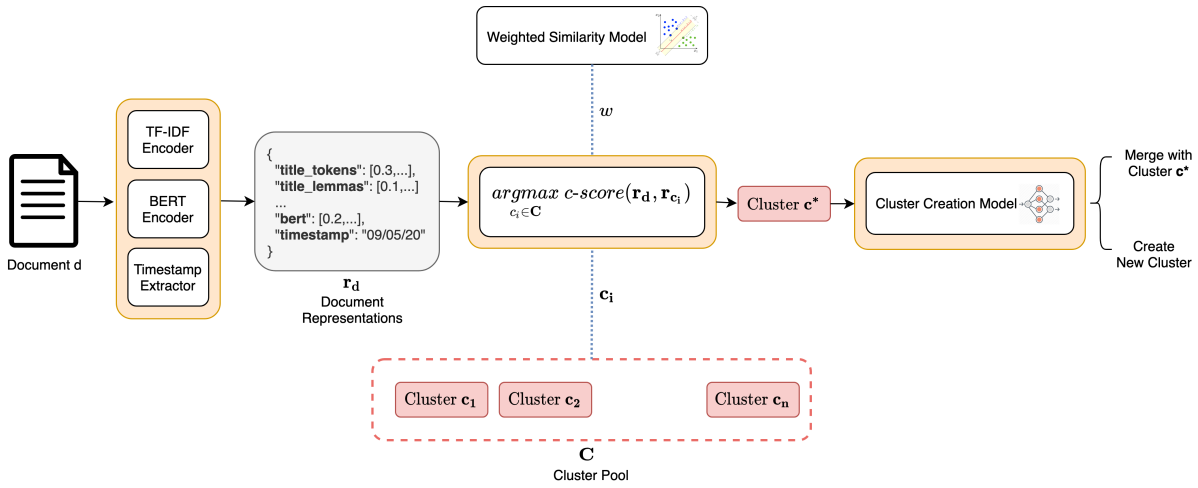

Figure 1: The architecture of the news stream clustering model, showing the clustering process for a single document in the news stream. At the end of the clustering process for each document, the cluster pool is updated based on the output from the cluster creation model, either by adding document d to cluster c^* or by creating a new cluster with the document.

3 Methodology

Our clustering model is a variant of the streaming K-means algorithm (MacQueen, 1967) with two key differences: (1) we compute the similarity between documents and clusters along a set of representations instead of a single vector representation; and (2) we decide the cluster membership using the output of a neural classifier, a learned model, instead of a static tuned threshold.

At any point in time t, let n be the number of clusters the model has created thus far, called the cluster pool. Given a continuous stream of news documents, the goal of the model is to decide the cluster membership (if any) for each input document. In our task, we assume that each document belongs to a single event, represented by a cluster. The architecture of the model, as shown in Figure 1, consists of three main components : (1) document representations, (2) document-cluster similarity computation using a weighted similarity model and (3) cluster creation model. In what follows, we describe each of these components.

3.1 Document Representations

Documents in the news stream have a set of representations, as shown in Figure 1, where each representation is one of the following types - sparse TF-IDF, dense embedding or temporal. We describe below these representation types and how clusters, which are created by our model, build representations from their assigned documents.

3.1.1 TF-IDF Representation

Separate TF-IDF models that are trained only on the tokens, lemmas and entities in a corpus are used to encode documents separately. For every document in the news stream, its title, body and title+body are each encoded into separate bags of tokens, lemmas and entities, creating nine sparse bag of word representations per document.

3.1.2 Dense Embedding Representation

Dense document representations are obtained by embedding the body of documents using BERT, with pre-trained BERT (P-BERT) without any fine-tuning as our baseline embedding model. In order to improve the effectiveness of contextual embeddings for our clustering task, we experiment with enhancements along two dimensions: (1) the fine-tuning objective, and (2) the provision of external knowledge. We train separate BERT models for (1) and (2) and use them to encode documents.

To evaluate the impact of the fine-tuning objective, we fine-tune BERT models on two different tasks - event classification (C-BERT) and event similarity (S-BERT). We also evaluate the impact of external knowledge on the embeddings through an entity-aware BERT architecture, which may be paired with either of the fine-tuning objectives.

Fine-tuning on Event Classification The goal of this fine-tuning is to tune the CLS token[1] embedding such that it encodes information about the

[1]The CLS token, introduced in (Devlin et al., 2019), is a special token added to the beginning of every document before being embedded by BERT

event that a document corresponds to. A dense and softmax layer are stacked on top of the CLS token embedding to classify a document into one of the events in the output space.

Fine-tuning on Event Similarity Fine-tuning on the task of event classification constrains the embedding of documents corresponding to different events to be non-linearly separable. Semantics about events can be better captured if the vector similarity between document embeddings encode whether they are from the same event or not.

For this, we adapt the triplet network architecture (Hoffer and Ailon, 2015) and fine-tune on the task of event similarity. Triplet BERT networks were introduced for the semantic text similarity (STS) task (Reimers and Gurevych, 2019), where the vector similarity between sentence embeddings was tuned to reflect the semantic similarity between them. We formulate the event similarity task, where the term similarity refers to whether two documents are from the same event cluster or not. In our task, documents from the same event are similar (with similarity = 1), while those from different events are dissimilar (with similarity = 0). Given the embeddings of an anchor document d_a, a positive document d_p (from the same event as the anchor) and a negative document d_n (from a different event), triplet loss is computed as

$$l_{triplet} = sim(d_a, d_n) - sim(d_a, d_p) + m \quad (1)$$

where sim is the cosine similarity function and m is the hyper-parameter margin.

Providing External Entity Knowledge In line with TDT's definition, entities are central to events and thus need to be highlighted in document representations for our clustering task. We follow Logeswaran et al. (2019) to introduce entity awareness to BERT by leveraging knowledge from an external NER system. Apart from token, position and token type embeddings, we also add an entity presence-absence embedding for each token depending on whether it corresponds to an entity or not. The entity aware BERT model architecture is shown in Figure 2. This enhanced entity-aware model can then be coupled with the event similarity (E-S-BERT) objective for fine-tuning.

3.1.3 Temporal Representation

Documents are also represented with the timestamp of publication. Unlike TF-IDF and dense embeddings, which are vector valued representations, the temporal representation of a document is just a single value (e.g "05-09-2020") which has an associated subtraction operation. The difference between two timestamps is defined as the number of intervening days between them. Section 3.2 describes how these timestamps are used for clustering.

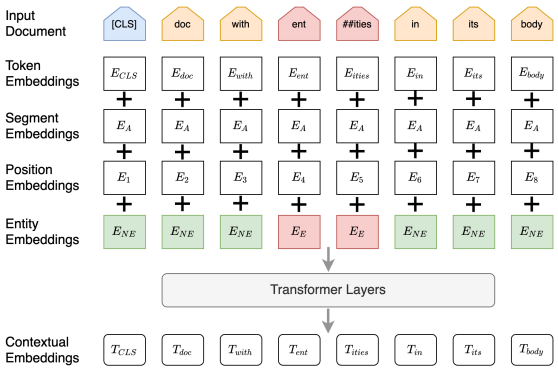

Figure 2: Entity-aware BERT model, with the additional entity presence (E_E) and absence (E_{NE}) embeddings

3.1.4 Cluster Representation

Since clusters are created and updated by our model, their representations need to be generated dynamically from the documents assigned to them. While documents in the news stream have a set of 11 representations (9 TF-IDF, dense embeddings and timestamp), clusters have two additional timestamp representations. Cluster representations are derived from documents in the cluster through aggregation. While dense embedding and sparse TF-IDF representations of a cluster are aggregated using mean pooling, clusters have three timestamp representations corresponding to different aggregation strategies - min, max and mean pooling.

3.2 Weighted Similarity Model

Once documents are encoded by a set of representations, they are compared to the clusters in the cluster pool to find the most compatible cluster. The similarity between a document and a cluster is computed along each representation separately and is then aggregated into a single compatibility score ($c\text{-}score$). While similarity along contextual embeddings and TF-IDF bag representations is computed using cosine similarity (as shown in Equation 2), timestamp similarity is computed using the Gaussian similarity function introduced in Miranda et al. (2018) (as shown in Equation 3).

Let $\mathbf{r_d}^v$ and $\mathbf{r_c}^v$ denote a dense or sparse vector

representation of a document d and cluster c respectively. Let $\mathbf{r_d}^t$ and $\mathbf{r_c}^t$ denote their timestamp representations. Let (i, j) correspond to a pair of document-cluster representations of the same type (as defined in Section 3.1). Document-cluster similarity is computed along each representation and aggregated using a weighted summation as

$$sim(\mathbf{r_d}, \mathbf{r_c}) = \{sim(\mathbf{r_d}^i, \mathbf{r_c}^j) \; \forall \; (i, j)\}$$

$$sim(\mathbf{r_d}^v, \mathbf{r_c}^v) = \frac{\mathbf{r_d}^v \cdot \mathbf{r_c}^v}{|\mathbf{r_d}^v||\mathbf{r_c}^v|} \quad (2)$$

$$sim(\mathbf{r_d}^t, \mathbf{r_c}^t) = e^{-\frac{((\mathbf{r_d}^t - \mathbf{r_c}^t) - \mu)}{2\sigma^2}} \quad (3)$$

$$c\text{-}score(\mathbf{r_d}, \mathbf{r_c}) = \sum_{(i,j)} w_j \cdot sim(\mathbf{r_d}^i, \mathbf{r_c}^j)$$

where μ and σ are tuned hyper-parameters of the temporal similarity function. It is noted here that since clusters have two additional timestamp representations, all three timestamp similarities are computed using the single document timestamp representation, as illustrated in Figure 3.

The dataset does not contain annotation for the degree of membership between a document and cluster and thus, the weights for combining the representation similarities can't be learned directly. To circumvent this issue, we train a linear model on a relevant task so that the trained weights can then be adapted to compute the compatibility score.

In our model, we train a linear model on a novel adaptation of the event similarity triplet loss used to train the S-BERT model. The triplet loss, as defined in Equation 1, can be adapted to a linear classifier if similarity has a related notion with regards to the classifier. SVM is an appropriate model since the degree of compatibility between a point x and a class c is given by the distance of the point from the class' decision hyperplane w_c. This distance, computed as $w_c \cdot x + b$, can thus be used as the similarity metric to adapt the triplet loss.

In our case, the inputs to the SVM model are vectors of document-cluster similarities along the set of representations $sim(\mathbf{r_d}, \mathbf{r_c})$. The adapted SVM-triplet loss is thus computed as shown below. Since we want to minimize this loss, we analyze its point of minima.

$$l_{svm-triplet} = w \cdot sim(\mathbf{r_a}, \mathbf{r_n}) - w \cdot sim(\mathbf{r_a}, \mathbf{r_p}) + m$$

$$l_{svm-triplet} = 0$$

$$\implies m = w \cdot (sim(\mathbf{r_a}, \mathbf{r_p}) - sim(\mathbf{r_a}, \mathbf{r_n}))$$

The adapted triplet loss can thus be modeled as a classification task with inputs $(sim(\mathbf{r_a}, \mathbf{r_p}) - sim(\mathbf{r_a}, \mathbf{r_n}))$ and the outputs m. For mathematical convenience, we set $m = 1$ without loss of generality. In this manner, we transform the event similarity triplet loss objective into a classification objective to train an SVM model. The novelty of this supervision is that we focus on learning useful weights and not a useful classifier. The learned weights, which minimize the event similarity triplet loss, are utilized for document-cluster c-score computation. During the clustering process, a document d is compared against all the clusters in the pool $\mathbf{C}$ to determine the most compatible cluster c^* as

$$c^* = \underset{c \in \mathbf{C}}{\arg\max} \; c\text{-}score(\mathbf{r_d}, \mathbf{r_c})$$

3.3 Cluster Creation Model

Since our clustering problem is non-parametric, each document in the stream could potentially be the start of a new event cluster. Thus, the most compatible cluster c^* might not actually be the cluster that the document corresponds to. Given a document and its most compatible cluster, the cluster creation model decides whether or not a new cluster is to be created. For this, we employ a shallow neural network which takes document-cluster similarities along the set of representations as input and decides if a new cluster should be created. Since the dimensionality of the input space for the network is small, we use a shallow network to prevent overfitting.

4 Experiments and Results

4.1 Data

To train and evaluate our clustering models, we use the standard multilingual news stream clustering dataset (Miranda et al., 2018), which contains articles from English, French and German. For our clustering task, we only use the English subset of the corpus, which consists of 20,959 articles. Articles are annotated with language, timestamp and the event cluster to which they belong, in addition to their title and body text. We use the same training and evaluation split provided by Miranda et al. (2018) and use the training set to fine-tune the parameters of the clustering model. The training and evaluation sets are *temporally disjoint* to ensure that the clustering models are tuned independent of the events seen during training.

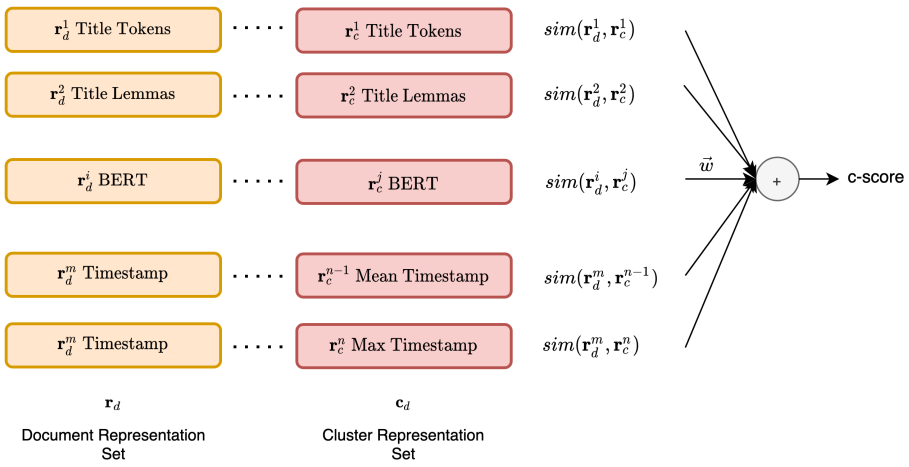

Figure 3: Computation of c-score: (a) similarities are computed for each representation individually using the appropriate similarity function (cosine or Gaussian); (b) subsequently, the computed similarities are aggregated into a single c-score value using the weights of the weighted similarity model (w)

4.2 Experimental Setup

We train our model pipeline in a sequence where each component model is supplied with the output from the component trained before in the sequence. For instance, the cluster creation model is trained using the embeddings from the fine-tuned BERT model and by selecting the most compatible cluster determined by the trained weighted similarity model. We experiment with multiple document representation sets, training all the component models each time and evaluating the entire clustering model on the test set.

We use the TF-IDF weights provided in the Miranda et al. (2018) corpus to ensure fair comparison with prior work. For training the event similarity BERT model (S-BERT), triplets are generated for each document using the batch-hard regime (Hermans et al., 2017) by picking the hardest positive and negative examples from its mini-batch[2]. We train the S-BERT model for 2 epochs using a batch size of 32, with 10% of the training data being used for linear warmup. We use Adam optimizer with learning rate $2e^{-5}$ and epsilon $1e^{-6}$. Document embeddings are obtained by mean pooling across all its tokens. For NER, we use the medium English model provided by spaCy (Honnibal and Montani, 2017).

Training instances for the weighted similarity and cluster creation models are generated by simulating the stream clustering process on the training set and assigning each document to its true event cluster. For the weighted similarity model, we generate triples of <document, true cluster, sampled negative cluster> and convert them into SVM training instances as mentioned in Section 3.2. Since all the training instances have the same label m, half the training set is negated to balance the dataset.

To generate training samples for the cluster creation model, the most compatible cluster is determined using the trained weighted similarity model for each document. A sample is then generated with input as the document-cluster similarities and output as 0 or 1 depending on whether the true cluster for that document is in the cluster pool or not. The dataset contains over 12k documents but only 593 clusters, entailing that the fraction of training samples where a new cluster is created is only 5%, making the dataset extremely biased. To mitigate this issue, we use the SVMSMOTE algorithm (Nguyen et al., 2011) to oversample the minority class and make the classes equal in size. For cluster creation, we train a shallow single layer neural network with two nodes using the L-BFGS solver (Nocedal, 1980). The weighted similarity and cluster creation models are trained using 5-fold cross validation to tune hyper-parameters and then on the entire training set using the best settings.

The clustering output is evaluated by comparing against the ground truth clusters. We report results on the B-Cubed metrics (Bagga and Baldwin, 1998) in Table 1 to compare against prior work.

4.3 Results

TF-IDF sets a tough baseline: Prior work has shown that sparse TF-IDF bag representa-

[2]We use the batch-hard implementation provided by Reimers and Gurevych (2019) at https://github.com/UKPLab/sentence-transformers

Model	Clusters Count (True Count - 222)	B-Cubed Metrics		
		Precision	Recall	F_1 Score
Laban and Hearst (2017)	873	94.37	85.58	89.76
Miranda et al. (2018)	326	94.27	90.25	92.36
Staykovski et al. (2019)	484	95.16	93.66	94.41
Linger and Hajaiej (2020)	298	94.19	93.55	93.86
Ours - TF-IDF	530	93.50	80.23	86.36
Ours - TF-IDF (out-of-order)	413	90.57	87.51	89.01
Ours - TF-IDF + Time	222	87.57	96.27	91.72
Ours - E-S-BERT	452	79.76	60.77	68.98
Ours - E-S-BERT + Time	471	92.70	74.69	82.73
Ours - TF-IDF + P-BERT + Time	196	83.12	**97.26**	89.63
Ours - TF-IDF + C-BERT + Time	321	83.10	91.33	87.03
Ours - TF-IDF + S-BERT + Time	247	88.30	96.10	92.04
Ours - TF-IDF + E-S-BERT	433	89.40	86.99	88.18
Ours - TF-IDF + E-S-BERT (out-of-order)	384	91.15	88.60	89.86
Ours - TF-IDF + E-S-BERT + Time	276	**94.28**	95.25	**94.76**

Table 1: Results of clustering performance for different document representation strategies as compared against contemporary models. P-BERT refers to pre-trained BERT; C-BERT refers to BERT fine-tuned on event classification S-BERT refers to BERT fine-tuned using triplet loss on event similarity; E-S-BERT refers to entity aware BERT fine-tuned on event similarity.

tions achieve competitive performance (Laban and Hearst, 2017; Miranda et al., 2018) and our experiments validate this observation. The clustering model that uses only sparse TF-IDF bags to represent documents achieves a very high score of 86.8% B-Cubed F_1 score, as shown in Table 1. If TF-IDF bags are used in combination with timestamps, then the performance further increases to 91.7%, setting a tough baseline to beat.

Contextual embeddings, by themselves, achieve sub-par clustering performance: In line with prior work, we observe that dense document embeddings, both when used as the sole representation and in conjunction with timestamps, are unable to match the clustering performance of TF-IDF bags. It can be seen in Table 1 that even our best BERT model (entity aware BERT trained on event similarity) only achieves an F1 score of 69% individually and 82.7% when combined with timestamp representations. These scores are 17.8% and 9% lower than their corresponding TF-IDF counterparts. BERT embeddings are richer representations that encode linguistic information including syntax and semantics through its pre-training. Thus, the model is unable to distinguish between events at the desired granularity and ends up clustering together topically related events (for instance, two different events related to soccer).

Fine-tuning objective impacts the effectiveness of embeddings for clustering: In most NLP tasks, fine-tuning contextual embeddings on a related pertinent objective is beneficial, we observe that the choice of fine-tuning objective is critical to the task performance. While the baseline pre-trained P-BERT model achieves a clustering score of 89.6% when used in conjunction with TF-IDF and timestamp representations (TF-IDF + P-BERT + Time), fine-tuning embeddings on event classification (TF-IDF + C-BERT + Time) drops the performance to 87%. This drop in performance can be attributed to the following reasons. Firstly, the large output space (593 events) and small dataset size (12k documents) make it hard for the model to learn effectively during fine-tuning. In addition to this, the classification objective requires that the embeddings of documents from different events be non-linearly separable. But this is not directly compatible with how the embeddings are used by the weighted similarity model, which is to compute cosine similarity. This discordance entails that the fine-tuning process degrades the clustering performance. The event similarity triplet loss is a more suitable fine-tuning objective and it is observed that fine-tuning BERT on this objective (TF-IDF + S-BERT + Time) results in a better clustering performance of 92.04%.

External entity knowledge makes embeddings more effective for clustering: The introduction of external knowledge through the entity aware BERT architecture significantly improves the clus-

Metric	TF-IDF + E-S-BERT + Time	Miranda	Gain
B-Cubed	94.76	92.36	$2.40^{\dagger}$
CEAF-e	76.93	69.57	$7.36^{\dagger}$
CEAF-m	93.31	90.19	$3.12^{\dagger}$
MUC	99.30	98.88	$0.42^{\ddagger}$
BLANC	98.13	96.93	$1.20^{\S}$
V Measure	97.98	97.01	$0.97^{\dagger}$
Adjusted Rand Score	96.26	93.87	$2.39^{\S}$
Adjusted Mutual Information	97.99	97.02	$2.97^{\S}$
Fowlkes Mallows Score	96.38	94.11	$2.27^{\S}$

Table 2: Results of clustering performance across different evaluation metrics. For each metric computed using precision, recall and F-1 scores, only the F-1 scores are reported. Statistically significant gains, with $p <<< 0.001$ are denoted by † and $p < 0.01$ by ‡. Gains denoted by $\S$ are not evaluated for significance, in line with literature.

tering performance of the model. It can be seen in Table 1 that introducing entity awareness and training on the event similarity task (TF-IDF + E-S-BERT + Time) results in a clustering score of 94.76%[3], achieving a new state-of-the-art on the dataset[4]. The results are statistically significant and p values from a paired student's t-test are reported in Table 2. This is almost 3 points better than the corresponding model without entity awareness, which highlights the importance of this external knowledge. When given external knowledge from an NER system, the BERT model, like sparse TF-IDF representations, is able to draw attention to entities and highlight them in the document embeddings. It is observed that the model learns to project entities and non-entities in mutually orthogonal directions and thereby adds emphasis to entities.

In our experiments, we observe an increase of almost 1 point in F_1 score by considering only a subset of the OntoNotes corpus (Weischedel et al., 2013) labels [5]. Ignoring entity classes like ORDINAL and CARDINAL helps as they don't provide useful information for our clustering task. The scores reported in Table 1 correspond to entity-aware models trained on this label subset. We also experimented with separate embeddings for each entity type instead of the binary entity presence-absence embeddings and observed that it degrades F_1 score by more than 2 points.

Ablating time and non-streaming input: When we ablate timestamp from the representation (rows that are not marked with "Time" in Table 1) and then stream documents in random order (rows marked with ("out- of-order") in Table 1), the number of clusters increase over when accounting for time. When ablating time, we also observe that supplying documents in random order produces fewer clusters and better b-cubed F_1 scores. We observe examples of clusters that are incorrectly merged in the absence of temporal information (in the out-of-order setting). See Appendix for actual examples from our output.

Cluster fragmentation is not captured well by B-Cubed metrics The improvements our model makes can be seen clearly by observing the number of clusters created by the model. While the previous state-of-the-art model produced 484 clusters, ours produces only 276[6], which is closer to the true cluster count of 222. Our model produces far less cluster fragmentation, resulting in a 79.4% reduction in the number of erroneous additional clusters created. We argue that this is an important improvement that is not well captured by the small increase in B-Cubed metrics.

While B-Cubed F_1 score is the standard metric reported in the literature, it is an article-level metric which gives more importance to large clusters. This entails that B-Cubed metrics heavily penalize the model's output for making mistakes on large clusters while mistakes on smaller clusters can fall through without incurring much penalty. In our experiments, we observed that this property of the metric prevents it from capturing cluster fragmentation errors on smaller events. In the news stream clustering setting, small events may correspond to recent salient events and thus, we want our metric

[3]The mean and standard deviation of the precision, recall and F-1 scores over five independent training and evaluations of our model are 94.64 ± 0.28, 94.72 ± 1.33 and 94.75 ± 0.59.

[4]We observe similar results on the TDT Pilot dataset (Allan et al., 1998), as shown in Section 4.4

[5]Our entity label subset consists PERSON, NORP, FAC, ORG, GPE, LOC, PRODUCT, EVENT, WORK OF ART, LAW and LANGUAGE.

[6]The mean and standard deviation of the cluster count over five independent training and evaluations of our model are 312 ± 27.

to be agnostic to the size of the clusters.

We thus use an additional metric that weights every cluster equally - CEAF-e (Luo, 2005). The CEAF-e metric creates a one-to-one mapping between the clustering output and gold clusters using the Kuhn-Munkres algorithm. The similarity between a gold cluster G and an output cluster O is computed as the fraction of articles that are common to the clusters. Once the clusters are aligned, precision and recall are computed using the aligned pairs of clusters. This ensures that unaligned clusters contribute to a penalty in the score and cluster fragmentation and coalescing is captured by the metric.

In order to ensure that our model's better performance is metric-agnostic, we also empirically evaluated our clustering model against prior work using several clustering metrics, the results of which are presented in Table 2. For this, we compare with Miranda et al. (2018) since their results are readily replicable. It can be observed that our model consistently achieves better performance across most metrics and is thus robust to the metric idiosyncrasies. Our model achieves an improvement of 7.36 points on the CEAF-e metric, which shows that our clustering model performs better than contemporary models on smaller clusters as well.

4.4 Results on TDT

To validate the robustness of our clustering model, we evaluate it on the TDT Pilot corpus (Allan et al., 1998). The TDT Pilot corpus consists of a set of newswire and broadcast news transcripts that span the period from July 1, 1994 to June 30, 1995. Out of the 16,000 documents collected, 1,382 are annotated to be relevant to one of 25 events during that period. Unlike the Miranda et al. (2018) corpus, TDT Pilot does not have the article titles. We, therefore, train all the components of our ensemble architecture using only the document body text. The TDT corpus does not provide pre-trained TF-IDF weights, so we train the weights on the entire corpus as a pre-processing step. Unlike Miranda, the TDT Corpus also lacks standard train and test splits. We create our own splits across 25 events. The splits are described and listed in the Appendix.

In line with our observations on the Miranda et al. (2018) corpus, we observe similar results on the TDT corpus. We achieve the best result on this corpus on a model with TF-IDF representations combined with temporal representations, BERT entity-aware representations fine-tuned on the event similarity task. The best result has a b-cubed precision of **81.62**, b-cubed recall of **95.89** and a b-cubed F_1 **of 88.18**. We generate **12 clusters** which matches the number of clusters in the ground truth.

We show that even in a cross-corpus setting, dense contextual embeddings, when augmented with pertinent fine-tuning, external knowledge and the conjunction of sparse and temporal representations, are a potent representation strategy for event topic clustering.

5 Conclusion

In this paper, we present a novel news stream clustering algorithm that uses a combination of sparse and dense vector representations. We show that while dense embeddings by themselves do not achieve the best clustering results, enhancements like entity awareness and event similarity fine-tuning make them effective in conjunction with sparse and temporal representations. Our model achieves new state-of-the-art results on the Miranda et al. (2018) dataset. We also analyze the problem of cluster fragmentation noting that our approach is able to produce a similar number of clusters as in the test set, in contrast to prior work which produces far too many clusters. We note issues with the B-Cubed metrics and we complement our results using CEAF-e as an additional metric for our clustering task. In addtion, we provide a comprehensive empirical evaluation across many metrics to show the robustness of our model to metric idiosyncrasies.

Acknowledgments

We thank Chao Zhao, Heng Ji, Rishita Anubhai and Graham Horwood for valuable discussions.

References

David Ahn. 2006. The stages of event extraction. In *Proceedings of the Workshop on Annotating and Reasoning about Time and Events*, pages 1–8, Sydney, Australia. Association for Computational Linguistics.

James Allan, Jaime G Carbonell, George Doddington, Jonathan Yamron, and Yiming Yang. 1998. Topic detection and tracking pilot study final report. In *Proceedings of the DARPA Broadcast News Transcription and Understanding Workshop*, pages 194–218.

Amit Bagga and Breck Baldwin. 1998. Algorithms for scoring coreference chains. In *The first international conference on language resources and evaluation workshop on linguistics coreference*, volume 1, pages 563–566. Citeseer.

Adham Beykikhoshk, Ognjen Arandjelović, Dinh Phung, and Svetha Venkatesh. 2018. Discovering topic structures of a temporally evolving document corpus. *Knowledge and Information Systems*, 55(3):599–632.

Federico Bianchi, Silvia Terragni, and Dirk Hovy. 2020. The dynamic embedded topic model. *arXiv preprint arXiv:2004.03974*.

Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008(10):P10008.

Bernhard E. Boser, Isabelle M. Guyon, and Vladimir N. Vapnik. 1992. A training algorithm for optimal margin classifiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, COLT '92, page 144152, New York, NY, USA. Association for Computing Machinery.

Robin Brochier, Adrien Guille, and Julien Velcin. 2020. Inductive document network embedding with topic-word attention. *Advances in Information Retrieval*, page 326340.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Adji B Dieng, Francisco J R Ruiz, and David M Blei. 2019a. Topic modeling in embedding spaces. *arXiv preprint arXiv:1907.04907*.

Adji B Dieng, Francisco JR Ruiz, and David M Blei. 2019b. The dynamic embedded topic model. *arXiv preprint arXiv:1907.05545*.

Pankaj Gupta, Subburam Rajaram, Hinrich Schütze, and Bernt Andrassy. 2018. Deep temporal-recurrent-replicated-softmax for topical trends over time. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1079–1089, New Orleans, Louisiana. Association for Computational Linguistics.

Alexander Hermans, Lucas Beyer, and Bastian Leibe. 2017. In defense of the triplet loss for person re-identification. *arXiv preprint arXiv:1703.07737*.

Elad Hoffer and Nir Ailon. 2015. Deep metric learning using triplet network. In *International Workshop on Similarity-Based Pattern Recognition*, pages 84–92. Springer.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

Thorsten Joachims. 2002. Optimizing search engines using clickthrough data. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '02, page 133142, New York, NY, USA. Association for Computing Machinery.

Kamrun Naher Keya, Yannis Papanikolaou, and James R. Foulds. 2019. Neural embedding allocation: Distributed representations of topic models. *arXiv preprint arXiv:1909.04702*.

Philippe Laban and Marti Hearst. 2017. newsLens: building and visualizing long-ranging news stories. In *Proceedings of the Events and Stories in the News Workshop*, pages 1–9, Vancouver, Canada. Association for Computational Linguistics.

Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ICML14, page II1188II1196. JMLR.org.

Mathis Linger and Mhamed Hajaiej. 2020. Batch clustering for multilingual news streaming. *arXiv preprint arXiv:2004.08123*.

Lajanugen Logeswaran, Ming-Wei Chang, Kenton Lee, Kristina Toutanova, Jacob Devlin, and Honglak Lee. 2019. Zero-shot entity linking by reading entity descriptions. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3449–3460, Florence, Italy. Association for Computational Linguistics.

Xiaoqiang Luo. 2005. On coreference resolution performance metrics. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 25–32, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

J. MacQueen. 1967. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*, pages 281–297, Berkeley, Calif. University of California Press.

Sebastião Miranda, Artūrs Znotiņš, Shay B. Cohen, and Guntis Barzdins. 2018. Multilingual clustering of streaming news. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4535–4544, Brussels, Belgium. Association for Computational Linguistics.

Elaheh Momeni, Shanika Karunasekera, Palash Goyal, and Kristina Lerman. 2018. Modeling evolution of topics in large-scale temporal text corpora. In *Twelfth International AAAI Conference on Web and Social Media*.

Hien M. Nguyen, Eric W. Cooper, and Katsuari Kamei. 2011. Borderline over-sampling for imbalanced data classification. *Int. J. Knowl. Eng. Soft Data Paradigm.*, 3(1):421.

Jorge Nocedal. 1980. Updating quasi-newton matrices with limited storage. *Mathematics of Computation*, 35(151):773–782.

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 and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Suzanna Sia, Ayush Dalmia, and Sabrina J. Mielke. 2020. Tired of topic models? clusters of pretrained word embeddings make for fast and good topics too! *arXiv preprint arXiv:2004.14914*.

Todor Staykovski, Alberto Barrón-Cedeño, Giovanni Da San Martino, and Preslav Nakov. 2019. Dense vs. sparse representations for news stream clustering. In *Proceedings of Text2Story - 2nd Workshop on Narrative Extraction From Texts, co-located with the 41st European Conference on Information Retrieval, Text2Story@ECIR 2019, Cologne, Germany, April 14th, 2019*, volume 2342 of *CEUR Workshop Proceedings*, pages 47–52. CEUR-WS.org.

Yee W Teh, Michael I Jordan, Matthew J Beal, and David M Blei. 2005. Sharing clusters among related groups: Hierarchical dirichlet processes. In *Advances in neural information processing systems*, pages 1385–1392.

Ralph Weischedel et al. 2013. *OntoNotes Release 5.0 LDC2013T19. Web Download*. Linguistic Data Consortium, Philadelphia.

Manzil Zaheer, Amr Ahmed, and Alexander J. Smola. 2017. Latent LSTM allocation: Joint clustering and non-linear dynamic modeling of sequence data. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 3967–3976, International Convention Centre, Sydney, Australia. PMLR.

Manzil Zaheer, Amr Ahmed, Yuan Wang, Daniel Silva, Marc Najork, Yuchen Wu, Shibani Sanan, and Surojit Chatterjee. 2019. Uncovering hidden structure in sequence data via threading recurrent models. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, WSDM 19, page 186194, New York, NY, USA. Association for Computing Machinery.

Deyu Zhou, Haiyang Xu, and Yulan He. 2015. An unsupervised Bayesian modelling approach for storyline detection on news articles. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1943–1948, Lisbon, Portugal. Association for Computational Linguistics.

A Appendix

The TDT corpus does not have a training and test split and we thus partition the corpus into two almost equal portions such that all documents in a single event are part of the same split. Our training set consists of 873 documents and our test set consists of 680 documents. The events in each partition of the TDT corpus is shown in Table 3

Events in Our Train Split
Karrigan Harding, Shannon Faulker, Quayle lung clot, Haiti ousts observers, NYC Subway bombing, Carlos the Jackal, USAir 427 crash, Lost in Iraq, Death of Kim Jong Il, Clinic Murders, Kobe Japan quake, Serbs violate Bihac, OK-City bombing
Events in Our Test Split
Pentium chip flaw, Cuban riot in Panama, Justice-to-be Breyer, Humble TX flooding, WTC Bombing trial, Cessna on White House, Aldrich Ames, Comet into Jupiter, Serbians down F-16, Carter in Bosnia, Halls copter, DNA in OJ trial

Table 3: Events in the training and test splits of the TDT Pilot corpus

Actual example of clusters incorrectly merged when documents are supplied out-of-temporal-order. Cluster label # 1024 in the Miranda testset, contains articles on Qatar being selected as FIFA worldcup host and issues with immigrant labour there are discussed in negative sentiment. The ground truth is a large cluster with 1869 documents. An example document title in this cluster is "Qatar World Cup sponsors targeted for improving workers' rights" with timestamp 2015-05-25 15:27:00. Cluster # 288 is a singleton about an upcoming Boston Celtics game and has a negative tone on their recent performance with an article titled "Celtics kick away a winnable game" with timestamp 2014-11-06 10:27:00. This is incorrectly merged with cluster # 1024. There are many more clusters that are incorrectly merged with cluster # 1024.