Unsupervised Text Segmentation via Deep Sentence Encoders: a first step towards a common framework for text-based segmentation, summarization and indexing of media content.

Unsupervised Text Segmentation via Deep Sentence Encoders[1]

IACOPO GHINASSI

Queen Mary University of London, CogSci, London, UK, i.ghinassi@qmul.ac.uk

In this paper we present a new algorithm for text segmentation based on deep sentence encoders and the TextTiling algorithm. We will describe how text segmentation is an essential first step in the re-purposing of media content like TV newscasts and how the proposed methodology can add value to other subsequent tasks involving such media products thanks to the features extracted for segmentation. We present experiments on Wikipedia and transcripts from CNN 10 news show and the results of the proposed algorithm will be compared to other approaches. Our method shows improvement over other unsupervised methods and it gives results that are competitive with supervised approaches without the need for any training data. Finally, we will give examples of how to re-purpose the encoded sentences, so to highlight the re-usability of the extracted sentence embeddings for tasks like automatic summarization**,** while showing how these tasks depend on the segmentation process.

CCS CONCEPTS • Applied computing~Document management and text processing~Document searching • **Computing methodologies~Artificial intelligence~Natural language processing • Information systems~Information retrieval~Retrieval models and ranking~Language models** • Applied computing~Arts and humanities~Media arts

Additional Keywords and Phrases: NLP, Linear Text Segmentation, Topic Segmentation, TV news broadcasts segmentation, Neural Sentence Encoders

1 INTRODUCTION

The possibility of re-using media products from different sources such as television, radio, etc. is very important for modern broadcasters, as users move more and more towards Internet-based, interactive platforms [1, 2]. These platforms facilitate the consumption of media contents in forms that are different from the original product: a portion of a news broadcast corresponding to a single story, for example, could be returned to a user in response to the users' query or detected interests. To do so, the programme would need to be divided into smaller units based on the topical content of such units [3].

[1] Proceedings of 2nd International Workshop on Data-driven Personalisation of Television (DataTV-2021) at the ACM International Conference on Interactive Media Experiences (IMX 2021), June 2021

Given a single, long text document like the transcript of a news broadcast, linear text segmentation, also referred to as topic segmentation, refers to the task of dividing it into smaller, topically coherent segments [4]. As mentioned, the task is a first and essential step for the retrieval of relevant information such as a single news story inside a newscast [5]. Similarly, the individuation of these segments is crucial for other applications like automatic summarization and discourse analysis [6].

Various techniques have been proposed during the years, both with the purpose of segmenting multimedia contents like news broadcasts1 [7, 8] or other contents such as business meetings [9] and newspaper articles [10].

Popular approaches include the use of lexical similarity [11], Hidden Markov Models [12, 13], Latent Dirichlet Allocation based topic models [14, 15] and Latent Semantic Analysis [16, 17]. More recent works have focused on supervised approaches with discriminative models like Support Vector Machines [18], Neural Networks [19, 20], conditional random fields [21] or some combination thereof [22].

While leading to better results, supervised approaches have the problem that they depend on the training data supplied and this often leads to problems of transferability of knowledge for the segmentation task, whereas supervised models might severely underperform in the case in which training data is not available for a specific domain [23].

In addition, the segmentation step is just the first of a larger pipeline that might include summarization, semantic search or segment labelling. Solutions based on topic modelling, for example, have the advantage over other task-specific approaches of providing additional, useful information at no additional cost for related, subsequent tasks like story units' tagging [14].

Given these considerations, this work proposes a simple, unsupervised approach that takes advantage of recent developments in transfer learning for NLP to obtain features for segmentation that can easily be re-

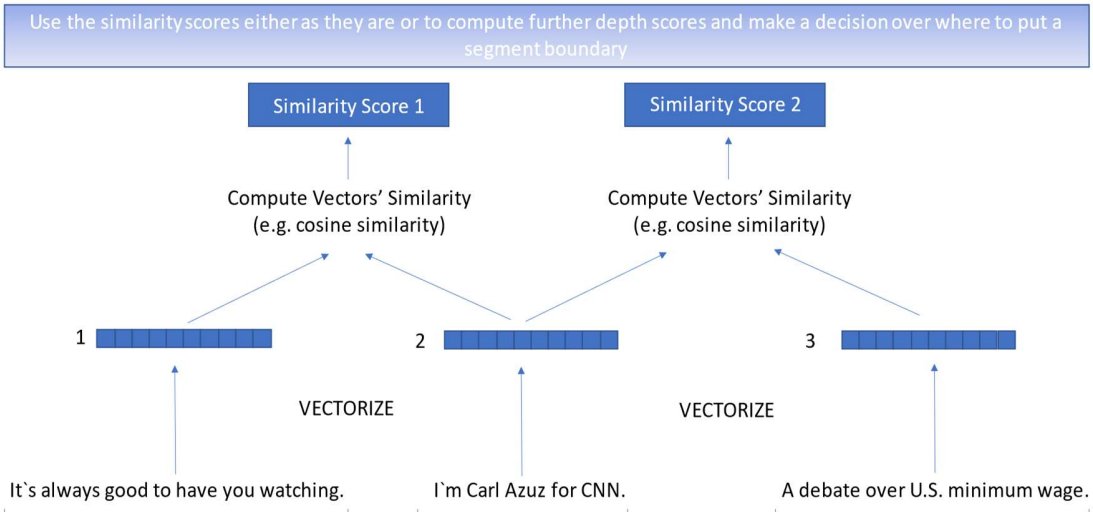

Figure 1: The basic structure of the TextTiling algorithm. In this case each "block of sentences" is represented by a single sentence. As it can be seen, the algorithm is quite flexible: any technique to convert a piece of text into a vector can work inside the algorithm (obviously affecting performance). The original algorithm transformed the text to vector through the so-called bag-of-words technique. The technique creates a vector with length equal to the number of words inside a prespecified vocabulary and fills it with the number of times each word appears in the given piece of text.

adapted for later uses. The next section introduces some relevant works in topic segmentation from which this research originated. We then present experimental results on two different datasets and, finally, we give an example of using our framework for segmentation and extractive summarization.

2 RELATED WORK

2.1 Linear Text Segmentation

One of the earliest approaches for linear text segmentation, TextTiling, pioneered the idea of using two adjacent sliding windows over sentences and comparing the two blocks of sentences inside these windows by means of cosine similarity between the relative bag-of-words vector representations [11]. Figure 1 gives a visual intuition of the algorithm. Applying the same algorithm with more informative representations of the blocks of sentences (the "vectorize" step in the figure) proved to improve performance [24]. To further mitigate the sparsity problem of the bag-of-words representation, generative topic models like Latent Dirichlet Allocation (LDA) [25] were soon adopted, first in probabilistic approaches involving dynamic programming [14, 26]. Topic modelling is a common task in natural language processing and (in the LDA perspective) it has the goal to represent documents as a mixture of a predefined number of topics. Drawing from both the topic modelling and lexical similarity perspectives, [15] proposed TopicTiling, an algorithm modelled after TextTiling but substituting the bag-of-words representations with denser topic vectors output by an LDA model. Another more recent strand of research uses vector semantics and, specifically, similarities between word vectors as a measure to determine the coherence of consecutive words. This concept has been variously applied either in combination with dynamic programming [27] or in more complex algorithms such as the one in [28], comparing consecutive sentences on the basis of a graph of similarities between their constituent words.

The systems described so far are unsupervised and were mainly tested on synthetic, text-only datasets such as the one proposed by [29]. At the same time, a parallel interest in topic segmentation for, especially, TV news broadcasts was fostered by events such as the DARPA sponsored TDT [30] and the TRECVID challenge [31]. In this context, many works proposed supervised systems exploiting the multimodal nature of TV broadcasts by using not just the transcript, but also audio and video information such as prosodic features and detected shot boundaries [32, 33].

In the supervised setting, the segmentation problem is treated as one of labelling individual units such as sentences, so as to individuate the unit where a segment ends [34] or starts [20]. Supervised systems have been shown to work better for segmentation than their unsupervised counterparts [19], especially when trained and evaluated on large text corpora: in such cases, the latest supervised, state-of-the-art systems show considerable improvement over previous approaches [35]. Whereas large annotated public corpora are not available the effectiveness of these models have been questioned, though [23].

In terms of datasets, different datasets have been used in the literature for evaluating linear text segmentation according also to the field from which the problem was approached. Among the first datasets specifically created for this purpose in the context of computational linguistics, [29] proposed a synthetic dataset created by randomly concatenating sections from different parts of the Brown Corpus. This dataset, however, has been shown to be too simple and not necessarily correlated with real word scenarios in which the segmentation systems might be deployed [28]. Starting from this observation, [19] proposed a dataset comprising 757,000 Wikipedia articles to overcome the limitations of previous datasets (especially their lack of

connection with real use case scenarios) and to provide a dataset big enough to train large, supervised models like neural networks.

For the more specific domain of TV newscasts segmentation, much fewer datasets have been developed. The most famous examples of such datasets are those released for the various editions of the TDT challenges [30] and two editions of the TRECVID challenge [31]. Generally, however, is quite common for authors to use private datasets created by the authors themselves [32] when experimenting on TV newscasts and TV programmes in general, which can be explained both with the fact that the TDT datasets must be paid for and that not many other datasets with the purpose of topic segmentation in TV programmes are publicly available.

2.2 Neural Sentence Encoders

As shown above, among unsupervised methods proposed for text segmentation, recently some literature has focused on the use of so-called word embeddings. Since the publication and release of the word2vec models [36] the Natural Language Processing community has gradually shifted towards using the word representations output by these models (i.e. word embeddings) as features for many tasks in the field. These models provide pre-trained, relatively dense[2] feature vectors corresponding to words in the given language, where these vectors have been optimized so that the vector representation of each word is geometrically closer to vectors of words it frequently appears with and distant from words it never appears with.

For many sentence-level tasks, it has been shown that the averaging of word vectors in a sentence can give a geometric representation of the sentence's semantics that is able to perform quite well in different tasks [37]. The introduction of contextualised word embeddings based on recurrent neural networks and, more recently, on Transformers proved to further improve performance on a variety of tasks, by creating different word vectors according to the surrounding words (i.e. context) of the target word [38]. Currently, what is probably the most well-known architecture of this kind is the Bidirectional Encoder Representations from Transformers (BERT), that reached state-of-the-art performance in many tasks at the time of its publication [39]. To further improve over the simple averaging or sum of word embeddings, specific neural sentence embedding algorithms have been proposed, such that they specifically output single embeddings for each input sentence. Among the most successful ones, the Universal Sentence Encoder [40] provides sentence embeddings from a deep averaging network [41] pre-trained on Wikipedia and the Stanford Natural Language Inference (SNLI) corpus [42]. More recent efforts adapted BERT to output sentence embeddings by using the distance between semantically related pair of sentences as an additional training objective [43]. Similarly, other Transformer-based architectures have been trained in the same way, while knowledge distillation has also been used to create sentence encoders that can encode the same sentence from different languages in a similar embedding space, therefore leading to single models that are able to perform well in multiple languages [44]. Especially, these latest transformer-based encoders have shown state-of-the art results in tasks ranging from semantic textual similarity to natural language inference. Moreover, recent research has shown how clustering these sentence embeddings can lead to effective topic modelling [45], leading to the consideration that they might be able to encode a change of topic in textual data such as TV newscasts.

[2] For comparison a common word vector's dimensionality is 300, while a bag-of-words vector will usually have much higher dimensionality (e.g. if we want to use a vocabulary of 10000 words, then the vector will contain 10000 dimensions).

Given these advances in sentence encoding and the fact that such methods, in our knowledge, have not been used in text segmentation and especially in the context of TV news broadcasts, this research addresses this gap in the literature by re-adapting a version of the TextTiling algorithm based on neural sentence encoders. In doing so, our main contributions are three:

1. We present a new algorithm that performs competitively with previous approaches and requires no additional data to be effective.
2. We evaluate the performance of different sentence encoders both in text segmentation and (briefly) in extractive summarization, informing future research on which neural sentence encoder might be more suited for these tasks.
3. We indicate how the features obtained from deep sentence encoders can be re-purposed and used for various different tasks related to the re-purposing and annotation of traditional media content like TV news shows.

3 METHODOLOGY

3.1 Algorithm

The proposed methodology closely follows the original TextTiling algorithm by [6]. Given the flexible nature of what can be included in the blocks to be compared inside the algorithm, various alternatives have in fact been proposed starting from this very same algorithm but using tf-idf weighting [24] or LDA [15]. This last approach demonstrated how the use of information more directly related to topic could dramatically improve the TextTiling approach. Recent work on neural-based sentence embeddings has shown how pre-training on multiple tasks deep neural networks can generate embeddings capturing lexical, discourse and topical structure [46]. This gives us a valid reason to experiment with some popular neural sentence encoders to obtain sentence representations to be compared in the TextTiling algorithm.

The general form of the algorithm is the same as TextTiling and its variants, but for the extraction of sentence embeddings and their use in computing the similarity scores between adjacent blocks. It consists in the following steps:

1. Extract sentences via a sentence tokenizer. In our case, we used the widely used and publicly available PUNKT tokenizer from NLTK python library [47].
2. For each sentence s_i, extract the relative embedding $e_i \in \mathbb{R}^n$ via the chosen sentence encoder, where n is the dimensionality of the numeric vector representing the sentence (i.e. the sentence embedding).
3. According to the chosen window parameters *w* compute $score(i)$ relative to sentence s_i as the cosine similarity between the average[3] of the embeddings in the two adjacent blocks of sentences having s_i as the rightmost sentence of the left block. Formally, for each position *i* we compute $score(i) = \frac{\boldsymbol{bl}(i)\cdot\boldsymbol{br}(i)}{\|\boldsymbol{bl}(i)\|\|\boldsymbol{br}(i)\|}$, where $\boldsymbol{bl}(i) = \frac{\sum_{n=i-w+1}^{i}\boldsymbol{e}_n}{w}$ and $\boldsymbol{br}(\mathrm{i}) = \frac{\sum_{n=i+1}^{i+w}\boldsymbol{e}_n}{w}$.
4. For each position *i* compute a depth score, as follow $ds(i) = \frac{1}{2}(score(l) + score(r) - 2score(i))$. In this context, $score(l)$ is found iteratively by comparing the scores on the left of $score(i)$ until a score at index *l* is found such that $score(l-1) < score(l) > score(l+1)$. The same is done for finding $score(r)$, whereas this time the peak is found on the right of $score(i)$.
5. If the number k of required segments is known, return the k boundaries having highest depth score. Else, return all boundaries that fall above a pre-defined threshold *p*.

[3] Different approaches have been experimented, such as max pooling, concatenation or more complex operations involving the embedding vectors, but the simple average proved to outperform other operators.

The algorithm per se is agnostic of what sentence encoder is used and, apart from the sentence encoder, it relies on just two hyperparameters, namely the window value w and (just if the number of segments is unknown) the threshold parameter p. Here, these two parameters and which sentence encoder works the best are found by optimising an objective metric on some held out data, but these parameters can also be pre-set on the basis of alternative considerations (e.g. runtime of the algorithm).

3.2 Sentence Encoders

The choice of the sentence encoder to be used is likely to have a strong effect on the performance of the proposed system. For this reason, we experimented with three different popular encoders all of which have their reported strength and weaknesses. Such encoders are:

- *Universal Sentence Encoder (USE)*: in 2018, Google Research released two task-agnostic sentence encoders under the name of universal sentence encoder [40]. Specifically, here we use just one of the two encoders that were released, namely the deep averaging network (DAN), further described in [41]. Despite the simplicity of this method, this encoder has proved to be quite effective, while not relying on the transformer architecture[4].
- *STSb-BERT base (SBERT)*: This sentence encoder is based on the base version of BERT [39] and improves over it by using additional training strategies so that the sentences that are supposed to be semantically similar have vectors closer to each other [43]. The resulting sentence embeddings outperformed previous sentence encoders (including universal sentence encoders) on the standard SentEval framework [48]. STSb-BERT is publicly available via the sentence_transformers python library released by UKP lab[5]. The same library has been used also for the third sentence encoder described below.
- *Paraphrase-xlm-r-multilingual-v1 (Para-xlm)*: This model derives from RoBERTa [49], a version of BERT having a different pre-training strategy that has been shown to make the model more robust and better than simple BERT in many tasks. The RoBERTa model is pre-trained on a dataset of paraphrases, then the number of its parameters are reduced by using knowledge distillation. This encoder has also the advantage of being able to produce embeddings for more than 50 languages thank to the additional knowledge distillation process applied to it and described in [44].

4 EXPERIMENTS

We test our approach in two datasets: one in the domain of TV news broadcasts and one more general, obtained by scraping Wikipedia articles. We evaluate segmentation performance with standard metrics, first, and then via an extrinsic, qualitative approach, wherein extractive summarization is applied to the extracted segments and the encoded sentences in the segments, so as to show a practical application that builds on top of the use of sentence encoders, while directly depending upon the quality of the segmentation.

4.1 Datasets

We present experiment on the following datasets:

- *wiki-50*: As mentioned above, [19] released a dataset specifically thought for text segmentation by scraping articles from Wikipedia. As the dataset is very big and it was proposed in the context of supervised text segmentation, a smaller dataset was also included by the authors to have a quicker way of comparison. This smaller dataset named wiki-50 is the first one we use here for evaluating the proposed approach and it consists in 50 Wikipedia articles, whereas just high-level section

[4] This also implies that the computation time of the encoding sentences is linear instead of quadratic.
[5] https://www.sbert.net/index.html

markers (i.e. heading 2 or lower) are considered to start different segments. Such dataset, even though outside the TV newscasts and, more in general, TV programmes domain, can give a useful indication about the performance of the segmentation algorithm in comparison with existent methodologies. It can also give an idea of how the algorithm might perform in cases such as Wikipedia articles, whereas a document generally follows a specific main topic, but it develops different related sub-topics at the same time[6].

- *CNN10*: For testing within the domain of TV newscasts, we developed a new, small dataset from the publicly available transcripts of the news broadcast show CNN10.[7] The dataset consists of 10 randomly picked shows from the programme and, as such, it can give an example of how an algorithm might perform with limited data, therefore further highlighting the advantage of our approach over supervised systems or LDA-based ones (for which an LDA model still needs to be fitted).

Table 1: Statistics of the datasets used.

Statistics	wiki-50	CNN10
Number of Segments	177	44
Average Segment Length	16 ± 3.47	20 ± 4.05
Segments per Document	3.40 ± 0.51	2.54 ± 0.49

5 INTRINSIC EVALUATION

The first type of evaluation of the model is intrinsic and quantitative. As usual for this kind of problem, the proposed method is applied to the two datasets and results are reported by using the metric described later in relation to the ground truth labels (i.e. the index of the last sentences for each segment).

Table 2 compares the configuration from our method leading to the best results with the performance of other methods from previous literature and naive baselines, all of which are further described in the next section.

As explained before, the presented method involves three parameters: the window value, the threshold value (in case the number of segments is not provided, as it is in our case) and the sentence encoder. A search in the parameter space has been done to find the optimal values for these parameters. In this case, the thresholds are calculated as $threshold(n) = x\sigma_{ds} + \mu_{ds}$, where x is the threshold multiplier found with the search, σ_{ds} is the standard deviation of the depth scores and μ_{ds} is their mean.

For consistency, in testing with all the methods we used a 5- fold Cross Validation, whereas the methods are tested on a concatenated fifth of the current dataset, while methods needing training or to fit additional models (e.g. LDA) are trained on the left-out data: the process is repeated five times for different partitions and the five results are then averaged.

5.1 Baselines

Apart from our method, table 2 reports the results of previous approaches from the literature and two naive baselines to better assess our approach. The methods being used for comparison are:

[6] A similar kind of document from the TV programme domain could be represented by documentary transcripts.
[7] http://transcripts.cnn.com/TRANSCRIPTS/sn.html

- *TextTiling* [11]: the original TextTiling algorithm. It uses a simple bag-of-words representation for creating the blocks to be compared in the same way as the algorithm described in the methodology section. As a pre-processing step, stop words were removed before applying this method.
- *TopicTiling* [15]: an alternative version of TextTiling using the topic ID assigned to each word with highest probability by an LDA model to build the sentence blocks. The LDA model is trained on the left-out data for each fold of the crossfold evaluation setting described in the previous section. Also in this case, stop words are removed before fitting.
- *Bi-LSTM* [19]: a hierarchical model comprising two bidirectional LSTM recurrent networks, one modelling the words' sequences in a sentence and one built on top of the first one and modelling the sentences. The model is supervised and it is trained to output a binary prediction for each sentence (boundary or not boundary).

We also include the performance of two naive baselines, to evaluate whether the approaches perform better than chance or than always outputting the majority class. These two baselines are:

- *Random Baseline*: using the same setting of [19] we include a random baseline that output a sentence-level decision over boundaries based on a binomial distribution having $\frac{1}{k}$ probability of success (i.e. a boundary is output), where k is the average length of segments in the dataset.
- *Zero Baseline*: We also included a baseline consisting in never predicting a segment boundary. This baseline helps better assessing the performance in terms of specific weaknesses of the used metric, as a method could artificially perform quite well according to them, but just as a result of tending never to predict a segment boundary (see next section).

5.2 Metrics

Early literature on text segmentation realised that traditional accuracy metrics might be too severe for evaluating the task: a model could place a boundary just next to the gold label, but metrics like accuracy or F1 would not take into account the fact that, for example, the method places boundaries closer to the actual ones when compared to other models [51]. Because of this, Pk metric was at first proposed by [10] as a metric specific for text segmentation and it defines the error as the probability of misclassifying two portions of a document (i.e. label them as coming from the same segment when they are not or vice versa). Pk values for the analysed methods on the two datasets are reported in table 2. It should be noted that for Pk the lower its value, the better. This metric requires a window value k to be specified: in this work we use half the average segment length for each dataset, as suggested by [10]. The metric, however, has been reported to penalise more false positives than false negatives, as, usually, the probability of not having a boundary is greater than having one (otherwise a boundary would be predicted mostly at every sentence) [50]. In this work we use this metric as it has been a popular choice for comparison in the past, but this evidence needs to be kept in mind when looking at the results and future research might consider different metrics to correct this bias.

5.3 Results

Table 2 shows the performance obtained with the best configuration from our approach compared with the previously described baselines. Specifically, the encoders leading to the best results are SBERT for the wiki-50 dataset and Para-xlm for CNN10. In both cases, the best window value is 15 and the best threshold multiplier is 1.5. For TextTiling and TopicTiling the reported results are the best ones obtained after having tuned their specific parameters, as well.

Table 2: Comparison of best results with our method and other models in terms of the described *Pk* metric. Best results are highlighted in bold. Asterisk indicates that the results have been reported from the original paper.

Method	wiki-50	CNN10
Our Method (best results)	27.01	**19.93**
TextTiling (best results)	35.35	32.34
TopicTiling (best results)	27.60	38.07
Bi-LSTM	**18.24***	41.44
Random Baseline	47.57	44.08
Zero Baseline	37.65	41.44

Our proposed method largely outperforms both the random and zero baselines and the unsupervised methods in the comparison (TextTiling and TopicTiling) for all the four best settings reported for the CNN10 dataset, while TopicTiling is competitive in the case of wiki-50 dataset (but ultimately, the best configurations of our approach perform better). It is interesting to notice how the performance of TopicTiling seems even worse than TextTiling in the small CNN10 dataset: TopicTiling requires an LDA model to be fitted to the data first and, as such, its quality is proportional to the quality of the LDA model, which performs much worse when the documents used to train it are not enough [51].

Bi-LSTM still performs considerably better in wiki-50, confirming the superiority of supervised approaches over unsupervised ones (or at least the present ones). The same method, however, performs quite poorly on CNN10. Specifically, the model performs the same as the zero baseline, as it never predicts a boundary. This is in line with the limitations of supervised methods observed by [23] in the context of text segmentation: when data are not enough to train a supervised system, the performance of such systems is going to decrease dramatically.

In general, our method seems to give good enough results, being competitive with supervised systems when enough data are available and better than them when training data is scarce, therefore proving to be particularly useful in such situations.

6 EXTRINSIC EVALUATION

In this section we present a qualitative extrinsic evaluation based on our algorithm. As shown in figure 2, the segment extraction is in fact just the first element of a variety of additional tasks that can be applied on the segments to obtain value from the processed media products. For this reason, we want to better contextualise our algorithm in such a scenario and see how the segmentation performance affects Extractive Summarization, i.e. the task of extracting the most relevant sentence(s) from a segment.

This task can be effectively tackled by using the same features used in our method for segmentation, i.e. the output of the sentence encoders. Our method, therefore, could be the starting point of potentially many tasks that rely both on a correct segmentation and on sentence encoders, saving time and resources while giving better results than other unsupervised approaches.

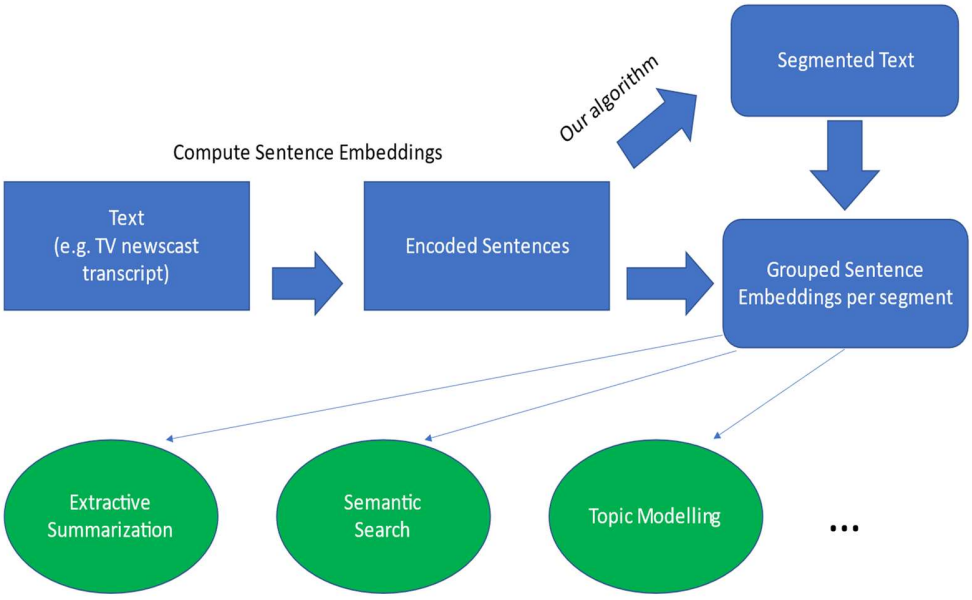

Figure 2: A diagrammatic depiction of where the segmenation process and, specifically, our approach stands in relation to other tasks that can bring value to, e.g., the online usage of media products.

6.1 Extractive Summarization

A useful thing for production teams and for users to have is some form of description of the media segments without having to watch or listen to them [1]. This can be obtained by using automatic text summarization techniques, a very active area of research in NLP [52]. Automatic summarization techniques usually take the form of extractive or abstractive techniques. Leaving aside abstractive techniques, extractive approaches automatically identify a number n of sentences ranked in order of semantic importance in the context of the document (in our case the segment). Even though extractive techniques usually perform worse than recent abstractive ones, they have the advantage that they might not need any training data, given a number of effective unsupervised approaches that have been developed for the purpose.

A document can be seen as a graph of sentences, whereas the semantic similarity of each sentence pair weights the edge between the two sentences. If a document is converted into a graph in this way, then a variety of algorithms from network analysis can be used to obtain the most central nodes in the graph, i.e. the most central sentences that, in turn, might give us the best summary of the document. One such algorithm, LexRank, was proposed by [53] and it involves the use of the degree of the inter-sentence cosine similarity matrix to compute a ranking of the sentences in a document. Having already a geometric representation of the sentences in a segment thanks to the sentence encoders from the segmentation algorithm, we can just apply the LexRank

algorithm on the extracted segments using the encoded sentences to annotate the segments themselves with a short summary.

Original Segment:
Hey, we have a newsletter and you`re going to love it. If you`re an educator or a parent looking for a preview on what`s on each day`s show, please visit CNN10.com and click on sign up for daily emails. You`ll get a sneak peak sent to your inbox every weeknight. It has been said that Border Collies are the smartest breed of dog. Debate that if you want but there is no debate that Wish and Halo, two Border Collies in California have just set a Guinness World Record for performing the most tricks by two dogs in one minute. How many did they do? Twenty- eight and separately one of them set a new record for the fastest five- meter crawl by a dog. Wish went the distance in just over two seconds.
Our model (encoder = USE):
Hey, we have a newsletter and you`re going to love it. If you`re an educator or a parent looking for a preview on what`s on each day`s show, please visit CNN10.com and click on sign up for daily emails. You`ll get a sneak peak sent to your inbox every weeknight. It has been said that Border Collies are the smartest breed of dog. Debate that if you want but there is no debate that Wish and Halo, two Border Collies in California have just set a Guinness World Record for performing the most tricks by two dogs in one minute. How many did they do? Twenty- eight and separately one of them set a new record for the fastest five- meter crawl by a dog. Wish went the distance in just over two seconds.
Our model (encoder = SBERT):
Hey, we have a newsletter and you`re going to love it. If you`re an educator or a parent looking for a preview on what`s on each day`s show, please visit CNN10.com and click on sign up for daily emails. You`ll get a sneak peak sent to your inbox every weeknight. It has been said that Border Collies are the smartest breed of dog. Debate that if you want but there is no debate that Wish and Halo, two Border Collies in California have just set a Guinness World Record for performing the most tricks by two dogs in one minute. How many did they do? Twenty- eight and separately one of them set a new record for the fastest five- meter crawl by a dog. Wish went the distance in just over two seconds.
Our model (encoder = Para-xlm):
Hey, we have a newsletter and you`re going to love it. If you`re an educator or a parent looking for a preview on what`s on each day`s show, please visit CNN10.com and click on sign up for daily emails. You`ll get a sneak peak sent to your inbox every weeknight. It has been said that Border Collies are the smartest breed of dog. Debate that if you want but there is no debate that Wish and Halo, two Border Collies in California have just set a Guinness World Record for performing the most tricks by two dogs in one minute. How many did they do? Twenty- eight and separately one of them set a new record for the fastest five- meter crawl by a dog. Wish went the distance in just over two seconds.

Figure 3: Text Segmentation performance on one passage from CNN10. The extracted segment is highlighted in red. Parameters used are window value = 5 and threshold multiplier = 1.

6.2 Segmentation Results

Figure 3 shows one segment in the dataset and the predicted boundaries from our method with the three different encoders, using fixed window and threshold value.

It can be noted how Para-xml and USE encoders both give a correct segmentation, while SBERT, in this case, includes an additional, unrelated sentence from the previous segment. In general, the segmentation performance seems acceptable also in the case of SBERT, as the boundary is placed extremely close to the correct one.

It is also evident that the performance of the algorithm on this particular example does not match which encoders were shown to work best in the intrinsic evaluation. This can partly be attributed to the metric itself (remembering the problems with the *Pk* metric), but also with the chosen example. For visualisation purposes, in fact, a short segment was chosen, but the average length of segments in the dataset is much larger (see table 1): this example, then, does not represent the average performance on CNN10 dataset.

This is also the reason why a small window value was chosen in this case, as a bigger one would have included multiple segments in the averaging of the encoded sentences, therefore probably leading to miss the segment boundary. Once more, then, this proves that the window value is a very important parameter in our method (and in all TextTilingbased approaches) and it is desirable to have previous knowledge of the domain or of similar ones, in order to tune that parameter.

Top Sentence (USE):

It has been said that Border Collies are the smartest breed of dog.

Top Sentence (SBERT):

Wish went the distance in just over two seconds.

Top Sentence (Para-xlm):

Twenty- eight and separately one of them set a new record for the fastest five- meter crawl by a dog.

Figure 4: Extractive summaries of the segment highlighted in figure 3, whereas a single sentence is extracted as a summary using LexRank. The three encoders give different results.

6.3 Summarization Results

After having obtained the segment as per figure 3, LexRank has been applied to extract a single sentence as a summary. Figure 4 shows which sentences the algorithm extracted for the three different encoders. Even though the segmentation results were the same for Para-xml and USE encoders, the extractive summarization algorithm gives different results in all three cases. By looking at the content of the sentences being extracted

under the light of whether they give relevant information to understand what the segment is about, we can see that Para-xml seems to return the most relevant sentence as it both includes the information of the event (i.e. the record) an the actors (i.e. the dogs) inside the sentence. USE comes second in the qualitative assessment, as it returns the first sentence of the segment, therefore highlighting that the segment is about Border Collies, but not giving any indication of the specific event. Finally, the sentence extracted via SBERT seems to be completely irrelevant. This evidence can be linked to the result from the segmentation step: because the segment includes an additional extraneous sentence, it is likely that the performance of the summarization is affected by that, as the extraneous sentence is taken into consideration when ranking sentences by their similarity with each other.

It can be noted that the algorithm for the extractive summarization runs in less than 1 second, as the sentence embedding were already computed for the segmentation task and the calculation of the cosine similarities between them takes considerably less time.

This example, then, shows how the performance of the summarization task depends on that of the segmentation one and how the proposed approach can help both the tasks via the re-purposing of the computed sentence embeddings.

7 CONCLUSION

This paper presented a novel method for linear text segmentation based on the TextTiling algorithm and the most recent advances in neural sentence encoders. The method proved to be effective and to lead to better results when compared to other unsupervised methods based on TextTiling, but it still performs worse than supervised approaches, when enough training data is available. Future work could investigate the use of these sentence representations in a supervised setting. Notwithstanding the segmentation performance, the proposed method proved to be beneficial also for additional tasks that might add value to a pipeline of re-purposing media products such as TV programmes for, e.g., online usage. In the present work, we experimented with extractive text summarization to show how once computed the sentence embeddings for text segmentation those same features can be re-purposed to annotate the segments with the most relevant sentence(s) from the segment itself. There are potentially many other use-cases in which the encoded sentences can be re-used, including semantic search and clustering-based topic models[8]. This work, then, represents a starting point for analysing the value that neural sentence encoders can add to text segmentation and, more specifically, to the segmentation and annotation of media products such as TV news broadcasts. Future research might broaden the scope of this paper by using more extrinsic evaluations and larger in-domain datasets.

Including information from the audio and video modalities is also a research direction worth pursuing. The proposed algorithm relies on good transcripts and, in the present form, it retrieves just the segments' text. To retrieve the segments in their original audio-visual format, additional steps would be needed such as alignment of audio and text and a shot boundary detection system to ensure that the video segments are cut appropriately. The features used for these post-processing steps might prove useful for the segmentation task itself.

REFERENCES

[1] Giuliano Armano, Alessandro Giuliani, Alberto Messina, Maurizio Montagnuolo, Eloisa Vargiu. 2011. Experimenting text summarization

[8] For a list of applications that use sentence encoders in an unsupervised fashion, see https://github.com/UKPLab/sentence-transformers/tree/master/examples/applications

on multimodal aggregation. In CEUR Workshop Proceedings.

[2] Fionn Murtagh, Adam Ganz, Joe Reddington. 2011. New methods of analysis of narrative and semantics in support of interactivity. Entertainment Computing 2, 2

[3] J. Reynar. 1999. Statistical models for topic segmentation. In Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics (ACL), 357–364

[4] Matthew Purver. 2011. Topic Segmentation. In Spoken Language Understanding (1st. ed.), Gokhan Tur and Renato De Mori (eds.). John Wiley & Sons, Ltd, Chirchester, UK, 291-317.

[5] Hermant Misra, François Yvon, Olivier Cappé, Joemon M. Jose. 2011. Text segmentation: A topic modeling perspective. Information Processing & Management 10, 4

[6] Marti A. Hearst. 1997. TextTiling: Segmenting text into multi-paragraph subtopic passages. Computational Linguistics 23, 1: 33–64.

[7] Camille Guinaudeau, Guillaume Gravier, Pascale Sébillot. 2012. Enhancing lexical cohesion measure with confidence measures, semantic relations and language model interpolation for multimedia spoken content topic segmentation. Computer Speech and Language 26, 2

[8] Imran Sehikh, Dominique Fohr, Irina Illina. 2018. Topic segmentation in ASR transcripts using bidirectional RNNS for change detection. In 2017 IEEE Automatic Speech Recognition and Understanding Workshop, ASRU 2017 – Proceedings.

[9] Matthew Purver, Konrad P. Körding, Thomas L. Griffiths, Joshua B. Tenenbaum. 2006. Unsupervised topic modelling for multi-party spoken discourse. In COLING/ACL 2006 - 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, 17-24.

[10] Doug Beeferman, Adam Berger, John D. Lafferty. 1999. Statistical models for text segmentation. Machine Learning 34, 1-3: 177–210.

[11] Marti A. Hearst. 1994. Multi-paragraph segmentation of expository text. In Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics (ACL).

[12] David M. Blei and Pedro J. Moreno. 2001. Topic segmentation with an aspect hidden Markov model. In Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 343–348.

[13] J. P. Yamron, I. Carp, L. Gillick, S. Lowe and P. van Mulbregt. 1998. A hidden Markov model approach to text segmentation and event tracking. In Proceedings of the IEEE Conference on Acoustics, Speech, and Signal Processing.

[14] Hermant Misra, Frank Hopfgarten, Anuj Goyal, P. Punitha, Joemon M. Jose, Susanne Boll, Qi Tian, Lei Zhang, Zili Zhang, Yi-Ping Phoebe Chen. 2010. TV News Story Segmentation Based on Semantic Coherence and Content Similarity. In Advances in Multimedia Modeling, 347-357.

[15] Martin Riedl and Chris Biemann. 2012. TopicTiling: A Text Segmentation Algorithm based on LDA. In Proceedings of the ACL 2012 Student Research Workshop i.

[16] Freddy Y. Y. Choi, Peter Wiemer-Hastings, Joanna Moore. 2001. Latent semantic analysis for text segmentation. In Proceedings of EMNLP.

[17] Yves Bestgen. 2006. Improving text segmentation using latent semantic analysis: A reanalysis of choi, wiemer-hastings, and moore. Computational Linguistics 32, 1.

[18] Maria Georgescul, Alexander Clark, Susan Armstrong. 2006. Word distributions for thematic segmentation in a support vector machine approach. In Proceedings of the 10th Conference on Computational Natural Language Learning (CoNLL-X), 101–108.

[19] Omri Koshorek, Adir Cohen, Noam Mor, Michael Rotman, Jonathan Berant. 2018. Text segmentation as a supervised learning task. In NAACL HLT 2018 - 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference.

[20] Pinkesh Badjatiya, Litton J. Kurisinkel, Manish Gupta, Vasudeva Varma. 2018. Attention-based neural text segmentation. In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics).

[21] Xiaoxuan Wang, Lei Xie, Mimi Lu, Bin Ma, Eng Siong Chng, Haizhou Li. 2012. Broadcast News Story Segmentation Using Conditional Random Fields and Multimodal Features. IEICE Transactions on Information and Systems E95 D, 5.

[22] Emiru Tsunoo, Peter Bell, Steve Renals. 2017. Hierarchical recurrent neural network for story segmentation. In Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH.

[23] Joe Barrow, Rajiv Jain, Vlad Morariu, Varun Manjunatha, Douglas Oard, and Philip Resnik. 2020. A joint model for document segmentation and segment labeling. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, 313–322.

[24] Michel Galley, Kathleen McKeown, Eric Fosler-Lussier, Hongyan Jing. 2003. Discourse segmentation of multi-party conversation. In Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL).

[25] Doug Blei, Andrew Y. Ng and Michael I. Jordan. 2003. Latent Dirichlet allocation. Journal of Machine Learning Research 3, 993–1022.

[26] Qi Sun, Runxin Li, Dingsheng Luo and Xihong Wu. 2008. Text segmentation with LDA-based Fisher kernel. In Proceedings of ACL-08, 269–272.

[27] Alexander Alemi and Paul Ginsparg. 2015. Text Segmentation based on Semantic Word Embeddings. arXiv. Retrieved March 11, 2021 from https://arxiv.org/pdf/1503.05543.pdf.

[28] Goran Glavas, Feerico Nanni, Simone Paolo Ponzetto. 2016. In *SEM 2016 - 5th Joint Conference on Lexical and Computational Semantics, Proceedings.

[29] Freddy Y. Y. Choi. 2000. Advances in domain independent linear text segmentation. In Proceedings of the 1st Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL).

[30] J. Allan, J. Carbonell, G. Doddington, J. P. Yamron, Y. Yang. 1998. Topic detection and tracking pilot study: Final report. In Proceedings of the 1998 DARPA Broadcast News Transcription and Understanding Workshop.

[31] Wessei Kraaij, Alan F. Smeaton, Paul Over. 2004. TRECVID 2004 - An Overview. In TRECVID 2004 Text Retrieval Conference TRECVID Workshop.

[32] Émilie Dumont and Georges Quénot. 2012. Automatic story segmentation for tv news video using multiple modalities. International Journal of Digital Multimedia Broadcasting.

[33] Bailan Feng, Zhineng Chen, Rong Zheng, Bo Xu. 2014. Multiple style exploration for story unit segmentation of broadcast news video. Multimedia Systems 20, 4.

[34] Jia Yu, Lei Xie, Xiong Xiao, Eng Siong Chng. 2018. An end-to-end neural network approach to story segmentation. In Proceedings - 9th Asia-Pacific Signal and Information Processing Association Annual Summit and Conference, APSIPA ASC 2017.

[35] Michal Lukasik, Boris Dadachev, Gonçalo Simões, Kishore Papineni. 2020. Text segmentation by cross segment attention. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, 4707–4716.

[36] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In Proceedings of NIPS'13.

[37] Tom Renter, Alexey Borisov, Maarten de Rijke. 2016. Siamese CBOW: Optimizing word embeddings for sentence representations. In 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016 - Long Papers.

[38] Noah A. Smith. 2019. Contextual word representations: A contextual introduction. arXiv. Retrieved March 11, 2021 from https://arxiv.org/pdf/1902.06006.pdf.

[39] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In Proc. of NAACL.

[40] Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, Ray Kurzweil. 2018. Universal Sentence Encoder. arXiv preprint arXiv:1803.11175.

[41] Mohit Iyyer, Varun Manjunatha, Jordan Boyd-Graber, Hal Daume III. 2015. Deep unordered composition rivals syntactic methods for text classification. In Proceedings of ACL/IJCNLP.

[42] Samuel R. Bowman, Gabor Angeli, Christopher Potts, 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, 632–642.

[43] 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), 3973–3983.

[44] Nils Reimers and Iryna Gurevych. 2020. Making monolingual sentence embeddings multilingual using knowledge distillation. arXiv. Retrieved 11 March, 2021 from https://doi.org/10.18653/v1/2020.emnlp-main.365.

[45] Dimo Angelov. 2020. Top2vec: Distributed representations of topics. arXiv. Retrieved 11 March, 2021 from https://arxiv.org/pdf/2008.09470.pdf.

[46] Sanjeep Subramanian, Adam Trischler, Yoshua Bengio, Cristopher J. Pal. 2018. Learning general purpose distributed sentence representations via large scale multitask learning. In 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings.

[47] Steven Bird and Ewan Klein. 2009. Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit. O'Reilly Media Inc.

[48] Alexis Conneau and Douwe Kiela. 2018. SentEval: An Evaluation Toolkit for Universal Sentence Representations. arXiv preprint arXiv:1803.05449.

[49] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. arXiv preprint arXiv:1907.11692.

[50] Maria Georgescul, Alexander Clark, Susan Armstrong. 2006. An analysis of quantitative aspects in the evaluation of thematic segmentation algorithms. In Proceedings of the 7th SIGdial Workshop on Discourse and Dialogue, 144–151.

[51] Jian Tang, Zhaoshi Meng, Xuan Long Nguyen, Qiaozhu Mel, Ming Zhang. 2014. Understanding the limiting factors of topic modeling via posterior contraction analysis. In 31st International Conference on Machine Learning, ICML 2014.

[52] Mahak Gambhir and Vishal Gupta. 2017. Recent automatic text summarization techniques: a survey. Artificial Intelligence Review 47, 1.

[53] Güneş Erkan and Dragomir R. Radev. 2004. LexRank: Graph-based lexical centrality as salience in text summarization. Journal of Artificial Intelligence Research, 22.