Computing and Informatics, Vol. 41, 2022, 78–97, doi: 10.31577/cai_2022_1_78

SEMANTIC SEGMENTATION OF TEXT USING DEEP LEARNING

Tiziano Lattisi

Department of Information Engineering and Computer Science
Università di Trento
38123 Trento, Italy
&
TxC2, Via Strada Granda
41 38069 Nago-Torbole (TN), Italy
e-mail: tiziano.lattisi@txc2.eu

Davide Farina, Marco Ronchetti

Department of Information Engineering and Computer Science
Università di Trento
38123 Trento, Italy
e-mail: davide.farina-2@studenti.unitn.it, marco.ronchetti@unitn.it

Abstract. Given a text, can we segment it into semantically coherent sections in an automatic way? Can we detect the semantic boundaries, if we know how many they are? Can we determine how many semantically distinct sections are in the text? These are the questions we address in this paper. To respond, we use the Bidirectional Encoder Representation from Transformer (BERT) to analyze the text and evaluate a function that we call local incoherence, which we expect to show maxima at the points where a semantic boundary is detected. Our results, although preliminary, are encouraging and suggest that our approach can be successfully applied. However, they are quite sensitive with respect to the text quality, as it happens in the case in which the text is derived from an audio stream via Automatic Speech Recognition techniques.

Keywords: Text segmentation, semantic boundaries, BERT

1 INTRODUCTION

Can a computer automatically split a text into semantically homogenous parts? The answer to this question is necessary, for instance, to automatically build an index for the text. When we search for information on paper, we know how to deal with written resources: traditionally we can use indexes, tables of content, and we are accustomed to visual hints like chapters, so that we can skim a book and detect its portions. For electronic resources, we use tools such as search engines to retrieve them, but when we obtain a document, to use it effectively we need to understand its structure (especially if it is long). If it is a video, we do not have powerful search tools.

We are presently witnessing a shift towards a society, where oral communication acquires a significance stronger than ever in the recent past. More and more we have audio and especially video resources (e.g., tutorials) which contain information or explain things. These resources can be transcribed, but once we do that, we are left with the problem of segmenting them into semantically homogenous parts if we want to obtain an index, which is usable for instance for providing a navigation tool for the resource. These are some of the reasons, why the question we pose is relevant.

1.1 Motivation

Our actual motivation to embark in this research is related to on-line learning and teaching. The recent COVID-19 pandemics has pushed towards distance education. Tools like Zoom and Microsoft Meet have been heavily employed to provide an instrument to continue providing instruction at all levels, from school to university, even during the lock-down. This happened both in synchronous and asynchronous mode. In synchronous mode, the video is used to mimic the usual lecture in class: students and teacher are present at the same time in a virtual space, and they can interact, even though in many case the traditional frontal lecture paradigm is applied. In asynchronous mode, lectures are recorded by the teacher, and made available for later use by the students. Very often, also synchronous lectures are recorded and made available for students.

Even before the pandemics, some lecturers were recording their live lessons, and provided them as videos to their pupils. As early as 2003, we demonstrated that such video-lectures are highly valuable for students [1], at least in the academic context, where often frontal lecturing is the only viable methodology, due to the large audiences. Later we suggested that the availability of recorded video-lectures can be used also to change the teaching paradigm [2].

In our view, the availability of video-lectures not only gives the obvious advantage of enabling students, who for any reason cannot attend at the “live” lecture to recover a lost lesson. What is most important, it provides the possibility of selectively reviewing material: this is precious for better understanding some difficult passages, for revising the learning material before an exam, for checking notes and for resolving discussions with peers.

Without a way of semantically annotating the video-lectures however, searching for the relevant passages may however prove to be a daunting task. It is therefore crucial to be able to provide ways, to effectively search through a single lecture or through a collection of them. Many techniques can be employed to this end. For instance, one could provide a lecture transcript where every word has an associated temporal marker and allow performing text searches: thanks to the temporal marker the result of the search would allow jumping to the video position, where the found term is used.

A stronger and more efficient way to find relevant passages would be to provide the possibility of performing semantic (rather than textual) search. A first step in this direction can be obtained for instance by exploiting the slides used in the lecture, at least in cases in which some presentation software such as e.g. Microsoft PowerPoint is used. From an analysis of the video stream, one can identify the slide changes, and from the slide titles one can obtain semantic information about the portion of the video associated with that particular slide. Title slides which separate sections in the lecture can provide even stronger indicators.

An explorable alternative is to use annotations provided by students, if the system allows them to add personal notes to the video streams, as discussed e.g. in [3].

Finally, another possible ingredient is starting from the lecture transcript to try to detect homogeneous semantic sections and label them. The first step in this direction is to find the boundaries between the semantic sections: this is what we address in the present paper.

1.2 The Research Questions

As we said, we want to find out if it is possible to automatically partition a text into subsections, which are semantically homogeneous. It is a task that may be not very easy even for humans. Different persons may detect different context boundaries, miss some of them, or locate their position in places that are different.

Our goal in the present work is to explore this problem, and to try to answer in at least a tentative way.

More precisely, we can pose the following two research questions:

RQ1: If we know that a text contains K context breaks, can we automatically detect them, or at least most of them?

RQ2: If we do not know how many context breaks are there, can we automatically discover how many they are, and correctly locate them?

This problem is a variant of text segmentation, a technique often used in natural language processing as first step towards e.g. text summarization, question answering and document noise removal.

In the rest of the paper, we will address these questions. In Section 2 we discuss the related work, in Section 3 the technique we propose and the dataset we employ for our test, in Section 4 we present our results, and finally in Section 5 we discuss and conclude.

2 RELATED WORK

Splitting text into subtexts is a common practice in Natural Language Processing and may serve vastly different goals. The very nature of the sought segments is also quite heterogenous: one may isolate symbols, words, phrases, groups of phrases, or higher-level structures like paragraphs, chapters or topics depending on what is the actual objective.

The ultimate goals are diverse: question answering e.g. [4], movie subtitling [5], extracting the introduction in podcast episodes [6], language identification [7], cross-lingual plagiarism identification [8], sentiment analysis e.g. [9, 10], summarization e.g. [11], clustering [12], story segmentation [13], topic partitioning [14, 15], image retrieval by their captions [16].

The segmentation can be unsupervised, or supervised e.g. [17]. Unsupervised methods are often based on heuristics, need extensive computational time and a huge amount of memory and are difficult to generalize, so that they are unpractical for real-world applications. Techniques used in unsupervised methods include, as indicated by Badjatiya et al. [18], lexical cohesion, statistical modelling, affinity propagation based clustering and topic modelling. Supervised methods employ decision trees and probabilistic models. They are usually costly, due to the needed human component.

Recently Pak and Teh [19] attempted to classify papers dealing with text segmentation techniques examining about 60 studies. Most of them were concerned with low level entities (characters, words, tags). 13 % dealt with phrases or sentences, and about one quarter of the total about higher level structures (text blocks, topics, subtopics). The dominating languages are English and Chinese (with more than 30 % each), and no paper was dealing with Italian (which is one of the two languages the present paper deals with, the second one being English). A work dealing with Italian text segmentation, which like the present one is based on BERT, is reported in [20].

3 TECHNIQUE

3.1 BERT

As a baseline for our work, we used Google's recent groundbreaking tool: the deep neural network architecture BERT (Bidirectional Encoder Representations from Transformers). Transformers are a novel neural network architecture based on a self-attention mechanism that is believed to be particularly well suited for language understanding and language translation.

Attention is a technique that attempts to mimic cognitive attention: it enhances the "important parts" of the input data while fading out the rest. Importance is given by the context, and its learning happens through minimization techniques. The notion of attention was first introduced by Badhanu et al. [21] to solve the problem of fixed-length context vector introduced in automated language translation based on encoders and decoders. The notion of attention has been applied also to the case of text segmentation (see e.g. Badjatiya et al. [18]).

Self-attention, also known as intra-attention, is a particular attention mechanism in which different positions of a single sequence are related to each-other in order to compute a representation of the sequence [22]. For example, in language translation, the meaning of a polysemous word obviously depends on the context. In broad terms, self-attention is a mechanism that allows discovering the true meaning by looking at the environment around the word. Obviously, this is of paramount importance in understanding queries formulated to a search engine.

BERT was created and made open source in 2018 by Devlin et al. [23] at Google. Since 2019, Google is leveraging it in user searches. BERT is a deeply bidirectional (someone says "non-directional"), unsupervised language representation. It is pre-trained using a plain text corpus composed of billions of words, hence learning a deep representation of natural language. An extra layer of domain specific training can be added if desired. Even if not having been specifically designed for that, BERT immediately achieved state-of-the-art performance on a number of natural language understanding tasks [20], outperforming many systems with task-specific architectures (neural or not neural, based on a sentence or paragraph embeddings), even though we are presently not able to fully understand why [24, 25]. In particular, BERT showed excellent performances on

- the GLUE benchmark [26], a collection of diverse natural language understanding tasks;
- SQuAD (both 1.1 and 2.0) [27], a collection of question–answer pairs;
- SWAG (The Situations With Adversarial Generations) [28], sentence pair completion examples that evaluate grounded commonsense inference.

BERT is extremely appealing due to its flexibility, and also because it is open-source and pretrained over a gigantic sample set. It can be used for sentiment analysis, semantic role labeling, sentence classification. In our case, we were interested in next sentence prediction, a task for which it is pretrained, and which consists in evaluating the probability that, given two phrases, the second follows the first. For instance, given the phrase "It is likely to rain", the following phrase "You'd better take an umbrella" makes sense, and hence it has a high probability to occur, while the phrase "You'd better stop eating cheese" would be evaluated with a rather low probability.

3.2 The Datasets

When we started working on our research questions, we were not aware of a standard dataset for this class of problems, and hence in first place we had to create one. There have been other datasets used in literature for segmentation tasks, but they are not standardized like it happens for other tasks, where challenges are defined.

For instance, Choi [29] used an artificial test corpus of 700 samples, each of which was a concatenation of ten text segments obtained by randomly select news document from the Brown corpus [30].

Badjatiya et al. [18] used three datasets. One was extracted from Wikipedia, and contained 300 randomly selected documents, each having an average segment size of 26. Another was composed of 227 chapters taken from medical books, for a total of 1 136 sections. A third one contained 85 fiction books, and the segment separation were the books' chapters.

Moreover, we were interested in applying our algorithm to the Italian language. Hence we defined our own datasets. They are small, since this work is preliminary. The datasets we built, in Italian and English, are:

- 6 datasets (4 in Italian, 2 in English) aggregating text from Wikipedia articles.
- 3 datasets composed of text of different news taken from on-line newspapers, in Italian.
- 2 datasets obtained from the transcription of video-lectures on Object Oriented Programming in Italian
- 10 datasets obtained from the transcription of video-lectures on Big Data in English.

The 4 Wikipedia datasets in Italian were composed in growing order of difficulty. Only the main text part of the Wikipedia article was kept, skipping the sections about related items, references etc.

The first sample contained 12 quite diverse arguments, as they were from very different domains (medicine, cinema, astronomy, . . .).

The second was a collection of articles all belonging from the same domain (Sport), but dealing with different sports (soccer, volley, rugby, . . .)

The third was about different forms of government (monarchy, dictatorship, theocracy, . . .)

Also the last sample was single-domain: it was a collection of Wikipedia pages about different Italian literary authors (Ugo Foscolo, Alessandro Manzoni, Giosuè Carducci, . . .).

The three newspapers datasets contained the same 13 news, extracted from one of the most popular Italian journals: "La Repubblica". The news was about 8 topics: politics, economics, soccer, cuisine, technology, medicine, science, crime. There was however some overlap even between some pairs of articles on different topics, as for example in some text economics overlapped with politics, and in other politics with crime. The difference among the three sets was the article arrangement: in the first case they were disposed so as to maximize the difference between subsequent topics (each segment was on a completely different topic than the next one), in the second we tried to minimize that (similar topics were disposed in adjacent positions) and the third one was an intermediate case, with a random arrangement.

The lecture transcriptions were from recorded video-lectures in Italian and in English. They all were about Computer Science university courses.

The two ones in Italian were obtained from the same lecture about Object Oriented Programming with JavaFX: one was the raw output of a standard Automatic Speech Recognition (ASR) engine, the second one was manually produced by refining and correcting such output. The text presented an extra difficulty, as the spoken language was Italian, but most technical terms were in English.

The samples from video-lectures in English were from various lectures of a single course about Big Data. The transcriptions were obtained by the ASR without further processing or corrections.

3.3 The Algorithm

As we mentioned, our starting baseline is BERT. In particular, we use the pretrained next sentence prediction. Given two arbitrary phrases, this gives an estimated probability p that, given two parameters, each one being a sentence, the second phrase follows the first (p is obviously a number between 0 and 1: close to 0 if extremely unlikely, 1 if virtually certain, i.e. the two phrases semantically correlate with each other, and are likely to be in a sequence in natural language). This can be easily achieved by using the transformer library [31].

One could hope in this way to find a semantic break in a text: by evaluating p for all pair of phrases in a text, discontinuities would be revealed by low values of p, hence allowing to break the text into semantically homogenous subsections. Of course, such view is highly naïve. A single phrase is not enough to delimitate a context, as it might be an incidental observation, or be a phrase that does not carry a particular semantic context. To hope to identify semantic boundaries, one has to extend the view to a further horizon.

We hence decided to work with larger blocks of text. We introduced multiple phrase correlations, in the following way.

Let us consider two adjacent blocks, each of N sentences, and call them clusters. On a semantic boundary, we expect that each of them has a high intracluster coherence, while the inter-cluster coherence should be low.

By intracluster coherence, we mean that the elements within a block k should give

$$P_{intra} = \frac{2}{N*(N-1)} \sum_{i \in set\ k} \sum_{j > i \in set\ k} P_{ij}.$$

close to 1, where i and j are the indexes of the phrases in block k and P_{ij} is the probability that phrase j follows phrase i. For instance, in a three-sentences block, we would evaluate P_{ij} for the pairs $(1, 2)$, $(1, 3)$ and $(2, 3)$. The factor in front of the sum is used to normalize the sum between 0 and 1. Of course, such definition is tweaking the meaning of the next sentence predictor, as it implicitly assumes that P_{ij} will be rather high if the two sentences are semantically related, even if they are not in a preceding-following relation.

Using the same tweaking of the concept, we define in a similar way the inter-cluster coherence. In this case we sum all the P_{ij} with i in the cluster k, and j in cluster $k+1$, for a total of N^2 pairs, and adjust accordingly the normalization.

$$P_{inter} = \frac{1}{N^2} \sum_{i \in set\ k} \sum_{j \in set\ k+1} P_{ij}.$$

Finally, our indicator, which we call the local incoherence at point k, becomes the intra-cluster coherences minus the inter-cluster coherence: $P_{intra} - P_{inter}$.

This means that if all the $N+N$ phrases are perfectly coherent, our indicator will be 0: maximum intra-coherence, but also maximum inter-coherence. The same will happen when all the phrases are uncorrelated, with in this case all coherence indicator close to zero. We hence expect local incoherence to peak when there are context changes: before and after the context break we are in coherent regions (and hence low values of local incoherence), but on the boundary we have high intra-coherence and low inter-coherence, so that our indicator becomes close to 1. Such expected behavior is shown in Figure 1.

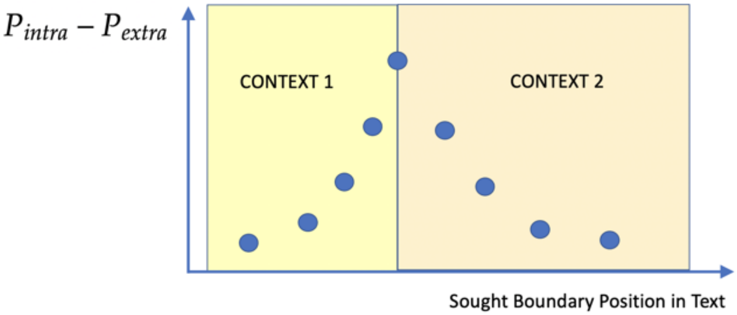


Figure 1. Example of the expected behavior of our indicator on a context boundary region

In principle the indicator could assume also negative values, and hence it is not strictly speaking a probability: this would happen in the odd case in which the inter-coherence is higher than the intra-coherence, a case which is rather odd and very unlikely to happen. It would mean that within the each of the two clusters there is very low correlation among the phrases, but that each phrase in the first block has a high correlation with phrases in the second block. Although such situation could be artificially created, at least up to a degree, it is extremely unlikely to "naturally" occur.

At this point, we perform the evaluation for every pair of adjacent blocks of dimension N, obtaining a plot that shows the local incoherence throughout the text, i.e. its value against the position where the local incoherence is evaluated.

The computational time needed to evaluate the local incoherence indicator grows quadratically on N. We limited ourselves to values of N between 3 and 6.

The resulting chart shows the value of local incoherence versus the positions in the text and suggests interesting results, but it is rather noisy, as shown in Figure 2, so we decided to try to filter out the noise.

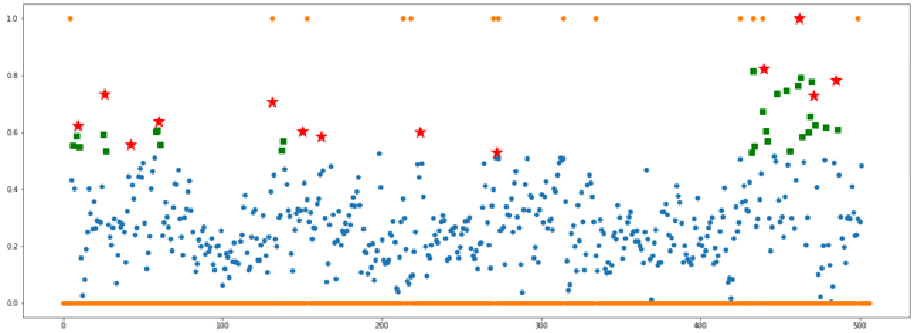


Figure 2. Example of result of the algorithm on a sample. Dots are the value of local incoherence. Stars indicate the local incoherence maxima, i.e. the predicted context breaks. The orange dots at the top indicate the actual position of the context changes.

To do that, we introduced a smoothing with a mobile average. This introduces an extra parameter w (w being the width of the mobile window). Within a window centered in position k, we evaluate the average of all the values between $k - w/2$ and $k + w/2$. The resulting value is assigned to the position k.

$$Smothed\ \delta_k = \frac{1}{w}\sum_{j=0}^{w-1}(P_{intra} - P_{inter})_{k-w/2+j}.$$

To be able to compute the mobile average, it is necessary to add a padding of $w/2$ zeroes at the beginning and at the end of the sequence of $P_{intra} - P_{inter}$ values. Such procedure essentially smooths the noise and helps to identify the truly relevant peaks.

Hence in our final plot we have on x the position where we calculate the value (i.e. the position k in the text), and on y the value of δ_k, i.e. the average of the local incoherence evaluated in the window of width w centered on x. An example of the result is in Figure 3. It shows the local incoherence plot for a sample, using $N = 3$ and $w = 7$.

The blue line in Figure 3 connects adjacent points to give a guide for the eye. The peaks indicate a maximum in the local incoherence, and hence show the pre-

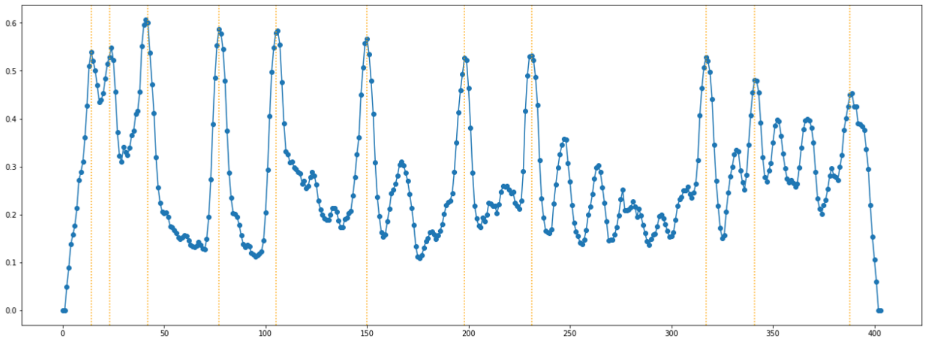


Figure 3. Example of result of the algorithm on a Wikipedia sample, using $N = 3$ and $w = 7$, plotting the smoothed local incoherence versus the position in the text where it is evaluated. The orange vertical lines indicate the position of the actual semantic boundaries.

dicted context boundaries. The orange vertical lines show the actual context boundaries. As it can be seen, in this sample all the highest peaks correctly identify all the context breaks. We will discuss the results in more detail in the next section.

Our code was written in Python, using TensorFlow [32] for the machine learning part and NumPy [33], Pandas [34] and Matplotlib [35] for data management. It was executed in a Google Colab [36] virtual machine.

4 RESULTS

In Figure 3 we have shown the results obtained by our algorithm on the first Wikipedia dataset. The 11 highest peaks in the graph of the local incoherence coincide exactly with the semantic boundaries, shown by the orange vertical lines. Hence the algorithm achieved its first goal: the 11 context boundaries were perfectly identified.

This was the luckiest case: not always the results were so good, as we will now see, but generally we do not go far from this quality level.

First of all, let us go back to our research questions.

RQ1: How can we correctly identify the semantic boundaries in a text, if we know how many they are?

Our response is: if we look for K semantic boundaries, we pick the K highest peaks in the local incoherence plot.

RQ2: How can we identify how many semantic boundaries are in a text, if we do not know how many they are?

> Our tentative answer is: let us fix a threshold value for the local incoherence: all peaks above the threshold are semantic boundaries, hence we just need to count how many they are.

As the data table will later show, our algorithm works generally very well on our dataset as far as RQ1 is concerned.

Unfortunately, instead, the answer to RQ2 depends on a parameter, the value of which is crucial for responding. The problem can be understood by checking the next example.

Even in an excellent case like the one we showed in Figure 3, not all peaks detect context boundaries: only the highest ones do that. If we know how many boundaries are there (as in RQ1, e.g. let us say N), the solution is simple: we select the highest N. If instead, as in RQ2, N is unknown, which peaks we keep as relevant, and which ones we drop? We need to define a threshold value, which we will call H. We will hence consider relevant only the peaks above H. Figure 4 helps understanding how critical the definition of such threshold is.

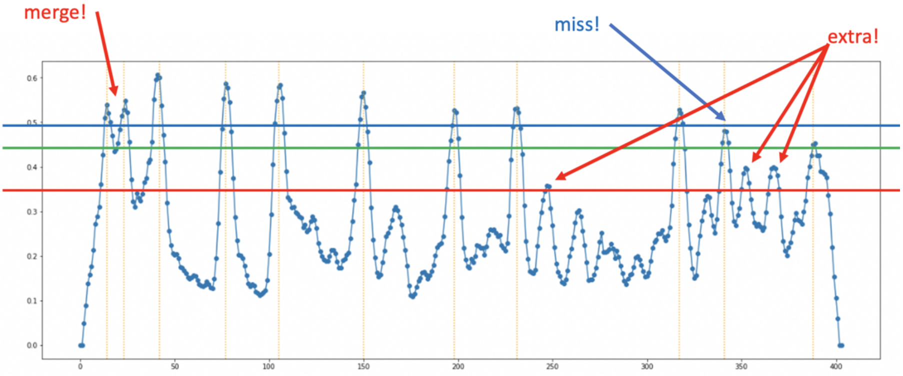


Figure 4. Exemplification of the problem of responding to RQ2. The horizontal lines (green, blue and red) correspond to different choices of the H parameter (see text for a more detailed explanation).

If we choose $H = 0.43$, as identified by the green line, the count number is correct. Even a slight variation in the value of H however changes the result. By increasing it by a tiny bit, the last peak would become irrelevant. The blue horizontal line exemplifies a choice ($H = 0.5$) such that two peaks are lost (on the right hand side).

On the other hand, by lowering H by a little with respect to the green line, the first two peaks would not cross the line in the transition from the first to the second: should then they be in considered as a single one? Although in the shown case one would be tempted to answer “no”, what if we have a twin peak with a tiny valley between them?

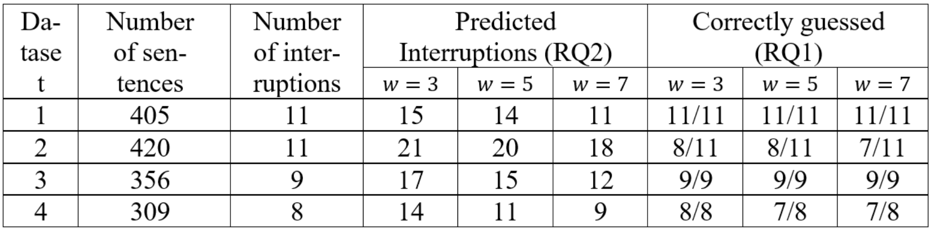
The red horizontal line shows the case $H = 0.34$ in which two peaks (on the left hand side) are "merged", and three new peaks (indicated in the figure as "extra") are introduced.

The answer to RQ2 depends hence in a critical way on the choice of H, and we could not to find a systematic way to identify it, so as to always have a satisfactory answer.

After this qualitative discussion, let us now see the data details. All the results we present were obtained with $N = 3$, which turned out to be sufficient to detect the semantic boundaries. As we mentioned, increasing N lets the computation time grow substantially. We experimented with larger values of N (up to $N = 5$) and since the results did not substantially change, we then felt unnecessary to employ larger values.

Table 1 presents the results for the Italian Wikipedia dataset. The average number of sentences per topic is a little less than 40.

For RQ1, the correctly identified semantic boundaries are reported in the last three columns, which differ in the value of the parameter w (width of the mobile average window). On the datasets 1 and 3 the precision always is 100 %, in the dataset 4 it is between 87.5 % and 100 % depending on the value of w. In dataset 2 instead the precision drops to a value between 64 % and 73 %, depending on w. It may be useful to recall the nature of dataset 3. It was about sports: the sport rules were described, and similar terms (like e.g. "team" or "ball") are present in most of them. Hence it is probably not surprising that this was the only dataset among the four for which we cannot claim full success on RQ1. However, even in this worst case the correctly found semantic boundaries reach 73 % with $w = 3$, while in all other cases the percentage is above 90 %.



Dataset	Number of sentences	Number of interruptions	Predicted Interruptions (RQ2)			Correctly guessed (RQ1)		
			$w = 3$	$w = 5$	$w = 7$	$w = 3$	$w = 5$	$w = 7$
1	405	11	15	14	11	11/11	11/11	11/11
2	420	11	21	20	18	8/11	8/11	7/11
3	356	9	17	15	12	9/9	9/9	9/9
4	309	8	14	11	9	8/8	7/8	7/8

Table 1. Results for the Wikipedia dataset (Italian language)

For RQ2, the number of predicted interruptions is generally higher than the actual value.

The dependency on w stems from the fact that the smoothing given by a larger window may hide features or introduce new ones. If we have two "true" peaks that are close to each other, a large moving average window may merge them in an intermediate position, hence hiding one of them and moving the position of the peak to a slightly incorrect place. Since for RQ1 we count the K highest peaks, where K

is the number of expected boundaries, a large value of w leads then to an incorrect identification of the relevant peaks. Also RQ2 is affected, since the number of peaks above the chosen threshold changes. This explains why by increasing w we get worse results.

Let us pass to the datasets, which contains text extracted from digital newspapers: as we already mentioned, the three dataset are compose by the same 13 news, and differ in the order in which the news are presented. The first arrangement attempts to maximize the semantic difference, avoiding putting news about the same general topic adjacent to each other, the second does the opposite, the third is in the middle, with a random positioning. Once again, the results for RQ1 are quite good, even if they show a decay when w grows (as we have also seen in the Italian Wikipedia dataset).

Also in this case, for RQ2, the number of predicted interruptions is generally higher than the actual value, even if in a less dramatic way, than in the previous table. Hence, we can claim that the results obtained in this case are coherent with what we observed in the Wikipedia dataset.

Dataset	Number of sentences	Number of interruptions	Predicted Interruptions (RQ2)			Correctly guessed (RQ1)		
			$w = 3$	$w = 5$	$w = 7$	$w = 3$	$w = 5$	$w = 7$
5	434	12	13	11	9	11/12	10/12	9/12
6	434	12	15	12	10	11/12	10/12	8/12
7	434	12	18	13	13	12/12	11/12	11/12

Table 2. Results for the news dataset (Italian language)

We then applied our machinery to a different domain: lecture transcriptions. This case differs in nature from the ones we have seen so far. In fact, in the above discussed cases, the semantic boundaries are well defined, being created by an artificial juxtaposition of different texts. In the lectures case, there is an overall coherence (the main topic being the lecture's argument) and a discourse flow. The detection of semantic boundaries is not so clearcut. To define them, we asked the teacher, whose lecture was recorded, to go through the transcript and mark the boundaries. There is a degree of arbitrariness in this: had we asked another person (e.g. a colleague or a student), the boundaries might have have been identified in different ways or in different locations.

The first lecture was a 1.5 hours university lecture in Italian about object oriented programming. It contained several technical terms, many of them in English. The spoken text was transcribed automatically, and we run our evaluation on it. The results (dataset 8) are much worse than what we found in the previous cases: the correctly found boundaries are only about 54 %.

Examining the transcripts, we noticed that there were many transcription errors. As we mentioned, the text was very technical and contained English words embedded

Da-tase t	Number of sen-tences	Number of inter-ruptions	Predicted Interruptions (RQ2)			Correctly guessed (RQ1)		
			$w = 3$	$w = 5$	$w = 7$	$w = 3$	$w = 5$	$w = 7$
8	529	13	20	15	9	7/13	6/13	3/13
9	508	13	18	15	9	9/13	8/13	4/13

Table 3. Results for the transcribed lecture (Italian language)

in Italian phrases. Moreover, the ASR had not been trained on the specific domain: it was hence not surprising that the transcripts were far from being perfect. Not only several terms were simply wrong (especially the interpretation of the English words intermixed in the Italian speech, but not only), but also some transcribed phrases made no sense at all.

We decided therefore to repeat the experiment after cleaning up the text. We manually corrected it, eliminating all the transcription errors. The number of sentences varies because the ASR defined sentence breaks also in places, where the manual correction removed the breaks.

The corrected transcript is our dataset 9. As it can be seen in Table 3, there is a clear improvement. The percentage of correctly identified boundaries jumps from 54 % to 69 %, coming close to what we obtained in the most difficult Wikipedia sample. However, we have to observe that about half of the semantic chunks were quite small: the last part of the lecture contained many quite short semantic segments, often only a few sentences long. It was in fact about short examples of different technical topics. The decay with w for RQ1, which is much stronger than that of the previous tables, probably stems from this fact.

We then used another dataset, using ASR transcriptions of other lectures on Big Data in English. In this case we had no corrections of the ASR output. Table 4 reports the results.

Da-tase t	Number of sen-tences	Number of inter-ruptions	Predicted Interruptions (RQ2)			Correctly guessed (RQ1)		
			$w = 3$	$w = 5$	$w = 7$	$w = 3$	$w = 5$	$w = 7$
10	1103	7	33	26	15	1/7	0/7	0/7
11	869	6	19	10	5	1/6	0/6	0/6
12	890	6	63	56	49	5/6	5/6	4/6
13	977	7	43	35	23	4/7	3/7	2/7
14	1064	6	62	53	37	5/6	4/6	6/6
15	1176	5	42	37	29	5/5	4/5	2/5
16	817	5	62	58	47	4/5	3/5	2/5
17	1008	4	67	62	46	3/4	3/4	3/4
18	1148	5	47	37	32	1/5	1/5	3/5
19	827	7	33	26	15	1/7	0/7	0/7

Table 4. Results for the transcribed lecture (English language)

Results were rather bad, in some cases even worse than those on Dataset 8. We were wondering, if this failure was due to the fact that we used the same trained network to deal with a different natural language (English instead of Italian). To check this hypothesis, we decided to repeat the test on an English text derived from Wikipedia, which we produced for this goal in a way similar to that we used for the Italian Wikipedia samples.

In this case results, reported in Table 5, were again positive: RQ1 has a 100 % precision with $w = 3$, and decreases a bit by enlarging the moving window.

Dataset	Number of sentences	Number of interruptions	Predicted Interruptions (RQ2)			Correctly guessed (RQ1)		
			$w = 3$	$w = 5$	$w = 7$	$w = 3$	$w = 5$	$w = 7$
10	477	11	12	10	8	11/11	10/11	8/11
11	584	7	12	9	6	7/7	6/7	5/7

Table 5. Results for Wikipedia (English language)

This result indicates that the problem is not the language, but rather the quality of text. Raw ASR, with spurious words and some meaningless sentence destroys the ability of the algorithm to detect the semantic boundaries. Unfortunately, manual correction of the ASR transcripts is very time consuming, so we were not able to correct the several hours of lectures on Big Data.

5 DISCUSSION AND CONCLUSION

Our work is preliminary in many senses. In the first place, the dataset we used is limited in the number of samples we used, and it is not standardized. However, we believe it is enough to hint that the technique we suggest can be effective and useful.

As we mentioned, the datasets we worked with were built on purpose, because there is not a standardized dataset in this domain, and because we wanted to work with the Italian language. Our datasets are limited in size, and certainly a more extensive work is needed to consolidate our results.

Also, our algorithm's inner working depends on three parameters: the size N of the chunks of text we compare to find semantic boundaries, the width w of the smoothing window and the threshold H used to determine how many boundaries are there.

RQ1 (identifying the position of the context boundaries if their number is known) depends only on the first two parameters, and we found that the relatively small values $N = 3$ and $w = 3$ were enough to obtain good results. In the presentation of our results, we already discussed the dependency on these two parameters. We claimed that a relatively small value of N, such as 3, is already enough: by making it bigger, computational time increases quadratically, while the obtained results are not improved. About the second parameter w, we argued that a certain amount of smoothing of the data produced by the algorithm is useful to filter the noise, but if the width of the window grows too much, some relevant features are lost, and some spurious ones may be introduced. The dependency on w is however not so dramatic to fully invalidate the results. The value $w = 3$ turned out to be optimal in our experiments.

This is confirmed also by the results of dataset 9 (corrected transcript of the Italian lecture), which suggests that short contexts are not detected. This is quite reasonable: if we sum the size N of the block and the width w of the window over which the local incoherence is averaged, we get a sort of minimum detectable block dimension. In our computation, such minimum block dimension was between 6 and 10, depending on the value of w (since we always had $N = 3$). Therefore, too short contexts were essentially invisible to our algorithm, and, as we mentioned, the last part of dataset 9 was characterized by rather short semantic segments, which might be one of the reasons why that sample was less successful than other ones.

RQ2 (identifying the number of the context boundaries) seems to be much more critical. In our approach, the answer to it depends on the third parameter, i.e. a threshold H above which we consider the peak in the local incoherence function to be significant. We were not able to find an optimal value for this parameter so that it could respond correctly to RQ2 by itself: we did not discuss in detail the attempts we made to optimize the threshold, since they were not convincingly successful and risked becoming an ad-hoc overfitting.

Does this limit the usefulness of our algorithm? If we think of a plausible usage scenario, the algorithm could be used to suggest places where semantic boundaries are likely to be, rather than a deterministic way to detect them. Considered as such, some overshooting in the response to RQ2, i.e. the inclusion of a certain number of false positives, is acceptable, as long as a subset of the identified locations actually detects true semantic boundaries. For instance, the indications provided by the proposed algorithm on a video transcript could be used to help detecting locations which are candidate for being a semantic boundary. These indications could be enriched by other heuristic signals such as e.g. frame changes, and a final decision could be (automatically) taken by considering all the information provided by the collective heuristics. Alternatively, these indications could be passed as they are to the final users ("user in the loop" technique). By observing the users behaviors (e.g. via learning analytics), or by using social techniques (such as occasionally asking users to declare if the obtained suggestion was good) one could then collect evidence helpful to classify the indications as true or false positives, refining in such a way the results.

The final point is the dependency of our results on the quality of the transcripts. We detected a relevant difference between a raw ASR transcription, and its manually corrected version. This fact should not be too surprising, thinking that a word representation in BERT depends on the context, in which the word is present (see e.g. [24]). Hence a (relatively) bad transcription can have dramatic effects on the obtained results. Therefore, if the source of the text is an audio, it is of paramount importance to have a good ASR transcriber. On the other hand, the same happens with humans: communication over a noisy phone connection remains intelligible if the information destroyed (or hidden) by the noise is little, and also depends on the importance of the missing pieces. Luckily the quality of ASR transcripts is increasing, and it is higher if the ASR is suitably trained on the specific domain: an operation that we did not perform.

In summary, we believe that, in spite of the limitations, we discussed its preliminary nature and this work suggests a promising road to tackle the problem of detecting semantic boundaries within a text.

REFERENCES

[1] Ronchetti, M.: Using the Web for Diffusing Multimedia Lectures: A Case Study. In: Lassner, D., McNaught, C. (Eds.): Proceedings of ED-MEDIA 2003 – World Conference on Educational Multimedia, Hypermedia and Telecommunications. Association for the Advancement of Computing in Education (AACE), 2003, pp. 337–340.

[2] Ronchetti, M.: Using Video Lectures to Make Teaching More Interactive. International Journal of Emerging Technologies in Learning (iJET), Vol. 5, 2010, No. 2, pp. 45–48.

[3] Ronchetti, M.—Lattisi, T.: Grab That Screen! Architecture of a System That Changes the Lecture Recording and the Note Taking Processes. In: Gennari, R. et al. (Eds.): Methodologies and Intelligent Systems for Technology Enhanced Learning, 9th International Conference (MIS4TEL 2019). Springer, Cham, Advances in Intelligent Systems and Computing, Vol. 1007, 2019, pp. 113–120, doi: 10.1007/978-3-030-23990-9_14.

[4] Oh, H. J.—Myaeng, S. H.—Jang, M. G.: Semantic Passage Segmentation Based on Sentence Topics for Question Answering. Information Science, Vol. 177, 2007, No. 18, pp. 3696–3717, doi: 10.1016/j.ins.2007.02.038.

[5] Scaiano, M.—Inkpen, D.—Laganière, R.—Reinhartz, A.: Automatic Text Segmentation for Movie Subtitles. In: Farzindar, A., Kešelj, V. (Eds.): Advances in Artificial Intelligence (Canadian AI 2010). Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 6085, 2010, pp. 295–298, doi: 10.1007/978-3-642-13059-5_32.

[6] Jing, Y.: Identifying Large-Scale Structures and Patterns in Narrative Data. Doctoral Dissertation, Indiana University, 2021.

[7] Fragkou, P.: Text Segmentation for Language Identification in Greek Forums. Procedia – Social and Behavioral Sciences, Vol. 147, 2014, pp. 160–166, doi: 10.1016/j.sbspro.2014.07.140.

[8] Ehsan, N.—Shakery, A.: Candidate Document Retrieval for Cross-Lingual Plagiarism Detection Using Two-Level Proximity Information. Information Processing and Management, Vol. 52, 2016, No. 6, pp. 1004–1017, doi: 10.1016/j.ipm.2016.04.006.

[9] Gao, Y.—Zhou, L.—Zhang, Y.—Xing, C.—Sun, Y.—Zhu, X.: Sentiment Classification for Stock News. 5th International Conference on Pervasive Computing and Applications (ICPCA), IEEE, 2010, pp. 99–104, doi: 10.1109/ICPCA.2010.5704082.

[10] Shi, H.—Zhan, W.—Li, X.: A Supervised Fine-Grained Sentiment Analysis System for Online Reviews. Intelligent Automation and Soft Computing, Vol. 21, 2015, No. 4, pp. 589–605.

[11] Liu, C.—Wang, Y.—Zheng, F.: Automatic Text Summarization for Dialogue Style. 2006 IEEE International Conference on Information Acquisition, 2006, pp. 274–278, doi: 10.1109/ICIA.2006.306009.

[12] Lamprier, S.—Amghar, T.—Levrat, B.—Saubion, F.: ClassStruggle: A Clustering Based Text Segmentation. Proceedings of the 2007 ACM Symposium on Applied Computing (SAC '07), 2007, pp. 600–604, doi: 10.1145/1244002.1244140.

[13] Xie, L.—Zeng, J.—Feng, W.: Multi-Scale TextTiling for Automatic Story Segmentation in Chinese Broadcast News. In: Li, H., Liu, T., Ma, W. Y., Sakai, T., Wong, K. F., Zhou, G. (Eds.): Information Retrieval Technology (AIRS 2008). Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 4993, 2008, pp. 345–355, doi: 10.1007/978-3-540-68636-1_33.

[14] Misra, H.—Yvon, F.—Cappé, O.—Jose, J.: Text Segmentation: A Topic Modeling Perspective. Information Processing and Management, Vol. 47, 2011, No. 4, pp. 528–544, doi: 10.1016/j.ipm.2010.11.008.

[15] Zhang, L.—Zhou, Q.: Topic Segmentation for Dialogue Stream. 2019 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC), 2019, pp. 1036–1043, doi: 10.1109/APSIPAASC47483.2019.9023126.

[16] Aung, N. M. M.—Maung, S. S.: Semantic Based Text Block Segmentation Using WordNet. International Journal of Computer and Communication Engineering, Vol. 2, 2013, No. 5, pp. 601–604, doi: 10.7763/IJCCE.2013.V2.257.

[17] Koshorek, O.—Cohen, A.—Mor, N.—Rotman, M.—Berant, J.: Text Segmentation as a Supervised Learning Task. Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL 2018), Vol. 2 (Short Papers), ACL, 2018, pp. 469–473, doi: 10.18653/v1/N18-2075.

[18] Badjatiya, P.—Kurisinkel, L. J.—Gupta, M.—Varma, V.: Attention-Based Neural Text Segmentation. In: Pasi, G., Piwowarski, B., Azzopardi, L., Hanbury, A. (Eds.): Advances in Information Retrieval (ECIR 2018). Springer, Cham, Lecture Notes in Computer Science, Vol. 10772, 2018, pp. 180–193, doi: 10.1007/978-3-319-76941-7_14.

[19] Pak, I.—Teh, P. L.: Text Segmentation Techniques: A Critical Review. In: Zelinka, I., Vasant, P., Duy, V., Dao, T. (Eds.): Innovative Computing, Optimization and Its Applications. Springer, Cham, Studies in Computational Intelligence, Vol. 741, 2018, pp. 167–181, doi: 10.1007/978-3-319-66984-7_10.

[20] Ranzato, P. L. R.: A Text Segmentation Technique Based on Language Models. Master Thesis, Politecnico di Milano, 2019.

[21] Bahdanau, D.—Cho, K.—Bengio, Y.: Neural Machine Translation by Jointly Learning to Align and Translate. Proceedings of 3rd International Conference on Learning Representations (ICLR 2015), 2016, arXiv: 1409.0473v7.

[22] Vaswani, A.—Shazeer, N.—Parmar, N.—Uszkoreit, J.—Jones, L.—Gomez, A. N.—Kaiser, Ł.—Polosukhin, I.: Attention Is All You Need. In:

Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (Eds.): Advances in Neural Information Processing Systems 30 (NIPS 2017), 2017, pp. 5998–6008.

[23] Devlin, J.—Chang, M. W.—Lee, K.—Toutanova, K.: BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding. 2018, arXiv: 1810.04805.

[24] Clark, K.—Khandelwal, U.—Levy, O.—Manning, C. D.: What Does BERT Look At? An Analysis of BERT's Attention. Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP, 2019, pp. 276–286, doi: 10.18653/v1/w19-4828.

[25] Kovaleva, O.—Romanov, A.—Rogers, A.—Rumshisky, A.: Revealing the Dark Secrets of BERT. Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), 2019, pp. 4365–4374, doi: 10.18653/v1/D19-1445.

[26] Wang, A.—Singh, A.—Michael, J.—Hill, F.—Levy, O.—Bowman, S.: GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP, 2018, pp. 353–355, doi: 10.18653/v1/W18-5446.

[27] Rajpurkar, P.—Zhang, J.—Lopyrev, K.—Liang, P.: SQuAD: 100 000+ Questions for Machine Comprehension of Text. Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, ACL, 2016, pp. 2383–2392, 2016, arXiv: 1606.05250.

[28] Zellers, R.—Bisk, Y.—Schwartz, R.—Choi, Y.: SWAG: A Large-Scale Adversarial Dataset for Grounded Commonsense Inference. Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2018, pp. 93–104, arXiv: 1808.05326.

[29] Choi, F. Y. Y.: Advances in Domain Independent Linear Text Segmentation. Proceedings of the 1st North American Chapter of the Association for Computational Linguistics Conference (NAACL 2000), ACL, 2000, pp. 26–33, arXiv: cs/0003083.

[30] Francis, W. N.—Kucera, H.: Brown Corpus Manual: Manual of Information to Accompany a Standard Corpus of Present-Day Edited American English for Use with Digital Computers. Brown University, Providence, Rhode Island, USA, 1979.

[31] Wolf, T.—Debut, L.—Sanh, V.—Chaumond, J.—Delangue, C.—Moi, A. et al.: HuggingFace's Transformers: State-of-the-Art Natural Language Processing. 2019, arXiv: 1910.03771.

[32] Abadi, M. et al.: TensorFlow: A System for Large-Scale Machine Learning. 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI '16), 2016, pp. 265–283.

[33] Oliphant, T. E.: A Guide to NumPy. Trelgol Publishing, USA, 2006.

[34] McKinney, W.: Pandas: A Foundational Python Library for Data Analysis and Statistics. Python for High Performance and Scientific Computing (PyHPC 2011), 2011.

[35] Barrett, P.—Hunter, J.—Miller, J. T.—Hsu, J. C.—Greenfield, P.: MatPlotLib – A Portable Python Plotting Package. In: Shopbell, P. L., Britton, M. C., Ebert, R. (Eds.): Astronomical Data Analysis Software and Systems XIV. ASP Conference Series, Vol. 347, 2005, pp. 91–95.

[36] Bisong, E.: Google Colaboratory. In: Bisong, E.: Building Machine Learning and Deep Learning Models on Google Cloud Platform: A Comprehensive Guide for Beginners. Chapter 7. Apress, Berkeley, CA, 2019, pp. 59–64.

Tiziano Lattisi has his Bachelor degree in mathematics, is Software Developer. He is founder and CTO of AXIA S.r.l. and of TxC2 S.r.l., a company working on an innovative videolecture system. He is also external consultant for the OECD (Organisation for Economic Co-operation and Development) and a collaborator of the University of Trento.

Davide Farina is Student of computer science at the University of Trento.

Marco Ronchetti is Professor in computer science at Universita di Trento. His background is in physics, and as a physicist he has been working for a year at IBM before starting his academic career. In the last 20 years, his main interest has been in the application of technology to teaching and learning.