A Flexible Architecture Using Temporal, Spatial and Semantic Correlation-Based Algorithms for Story Segmentation of Broadcast News

Alberto Palomo-Alonso, David Casillas-Pérez, Silvia Jiménez-Fernández, Jose A. Portilla-Figueras, and Sancho Salcedo-Sanz

***Abstract*—In this article, we propose a novel flexible architecture, with different algorithmic procedures, for effective story segmentation of broadcast news from subtitle files. The proposed system exploits spatial and temporal distance, as well as sentence similarity, to classify different stories in news broadcasts. The computational algorithms which form the architecture mainly focus on each sentence's features (temporal distance, spatial distance, and semantic similarity), and are combined to build an overall classifier. The first algorithm in the architecture focuses on the segmentation task, detecting boundaries between news. The second and third algorithms identify high semantic correlation between pieces of text, whether they are consecutive in space or not. Video Text Track (VTT) subtitle files are used to evaluate the performance of the proposed approach, although any file format that includes temporal information could also be considered. These VTT files may contain text errors and inaccuracies, and the proposed algorithms have been designed to deal with noisy content.**

***Index Terms*—Natural language processing, correlation matrix, BERT, video text track.**

I. Introduction

A HUGE number of newscasts are broadcasted every day around the world. Retrieving exact information from massive newscast databases has become a crucial problem for users, where the only practical solution is to rely on automatic segmentation processes. Thus, Story Segmentation of Broadcast News (SSBN) has become a critical problem in information retrieval. Given a stream of text transcriptions of a broadcast audio/speech/video to SSBN, it identifies the locations where the news or topics change [1].

Manuscript received 22 July 2022; revised 22 February 2023 and 10 July 2023; accepted 26 July 2023. Date of current version 11 August 2023. This work was supported in part by Universidad de Alcalá - ISDEFE Chair of Research in ICT and Digital Progress, and in part by the Spanish Ministry of Science and Innovation (MICINN) under Project PID2020-115454GB-C21. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Kai Yu. *(Corresponding author: Silvia Jiménez-Fernández.)*

Alberto Palomo-Alonso, Silvia Jiménez-Fernández, Jose A. Portilla-Figueras, and Sancho Salcedo-Sanz are with the Department of Signal Processing and Communications, Universidad de Alcalá, 28805 Madrid, Spain (e-mail: a.palomo@uah.es; silvia.jimenez@uah.es; antonio.portilla@uah.es; sancho.salcedo@uah.es).

David Casillas-Pérez is with the Department of Signal Processing and Communications, Universidad Rey Juan Carlos, 28942 Madrid, Spain (e-mail: david.casillas@urjc.es).

Digital Object Identifier 10.1109/TASLP.2023.3301231

Usually, broadcast news programs consist of multiple stories, put together to form the newscast program. Manual segmentation of newscast programs by a human expert is, of course, possible and accurate, but it is a very costly task, both in terms of time and effort [2]. Automatic story segmentation models can handle entire news broadcasts for most languages, and get automatic text translation from them. However, currently, these models do not reach a high performance as a human could do manually, resulting in slightly noisy results. Obtaining improvements in the development of automatic models for accurate SSBN is therefore a hot topic in the speech and text recognition field.

Several efficient automatic approaches have been proposed before for SSBN tasks. Less than a decade ago, the first models, now deprecated, proposed the use of Universal Sentence Encoder (USE) [3] and Word to Vector (W2V) [4] as segmentation algorithms baseline [1], [5], [6], [7], [8]. More recently, new and groundbreaking language models such as Bidirectional Encoder Representation from Transformers (BERT) [9] were introduced, developing more efficient algorithms for SSBN problems [10]. Some of the previous approaches considered time-based modeling [2], which makes use of the spatial distance feature of the text. Some of these models needed to make some assumptions to obtain good results, such as prior sentence clustering [11] or a prior estimation of the number ofpieces of news to be segmented [14]. In general, such assumptions cannot be made for SSBN segmentation tasks, as the only information available is obtained from Automatic Speech Recognition (ASR) modules. We can also find other approaches for SSBN based on Deep Learning (DL) models [12], [13], [14] or statistical models [15] which achieve good performance in some specific SSBN problems. Even though advanced DL techniques reach efficient solutions, the BERT approach is still the most efficient, flexible, and leading-edge model for Natural Language Processing (NLP) tasks, as shown in different recent works [10].

In this article, we propose a novel architecture for SSBN segmentation problems, which considers the three main features of any newscast to perform an accurate segmentation: time information, spatial distance, and semantic correlation among news. It is important to highlight that the algorithms implemented do not take into account any assumptions, such asknowing the number of news in a newscast [2] nor include manual sentences pre-clustering to initialize the algorithms [11]. The proposed architecture not only includes novel algorithms to perform each task but also allows implementing any other custom segmentation and classification algorithm (deep learning, statistical or classic) and a *Specific Language Model* (SLM) as a baseline, making the architecture highly flexible. For this purpose, the system'sinputs are subtitle files, in our case Video Text Track (VTT) subtitle files, although any file format that includes temporal information could be considered. These subtitle files contain timestamps as a temporal reference for each generated text, arranging the time and text information as a set. One or more sets define a piece of news. These VTT files may include text errors and inaccuracies, and the proposed algorithms have been developed to deal with noisy content. The algorithms proposed in this architecture are unsupervised approaches, which prioritize andsupport the flexibility and generalization of the architecture. Adding supervised models to a flexible architecture is possible, but has some issues, such as extrapolating the architecture to other problems or fields results in a process of "fine-tuning" of the algorithms' parameters, which requires a specific training database with a specific ground truth. Note also that two previous works have been used as part of the proposed modules of the whole architecture. The first work consists of a module that we have called Gaussian Proximity Adapter (GPA), which is a slight modification of the work in [2]. The second work consists of BERT [9], a transformer-based language model. We implemented a community pre-trained model [16] to obtain vector representations of sentences in the news. Apart from these two modules, the rest of the architecture is completely newly proposed in this work: two algorithms (one for matrix segmentation and one for clustering), and the architecture where these algorithms are implemented have been proposed in this article.

To achieve the news segmentation and classification, we propose a sequential architecture, that works on a single subtitle file, considering different data features separately. The first part of the proposed architecture is the *Temporal Distance Manager* (TDM), which is specific to databases with temporal information. The inputs to this module are timestamped blocks of text containing the news information provided by the Automatic Speech Recognition module that generates the VTT files. Then, the text contained in these news information blocks is analyzed by the Specific Language Model (SLM), generating a semantic correlation matrix between the blocks. For this purpose, the SLM chosen has been BETO [17], the Spanish adaptation of the BERT Language Model [9]. The TDM also looks for temporal gaps in the subtitle file to include this information in the correlation matrix. The second part of the architecture is the *Spatial Distance Manager* (SDM), which considers the number of sentences between text blocks in the subtitle files. Finally, the last module, called *Later Correlation Manager* (LCM), looks for highly semantically correlated pieces of text using iterative calls to the SLM. In this work, we also propose two novel algorithms that provide a high segmentation performance: The first algorithm, called *Proximity-Based Merging by Means* (PBMM), uses time, space, and semantic information to identify boundaries corresponding to different pieces of news within the newscast raw text. The second algorithm, called *Favorite-Based Bidirectional Connection Merging* (FBBCM), merges two or more text blocks if they are likely to belong to the same piece of news.

The proposed architecture has been tested in the segmentation of different Spanish Television (RTVE) [18] newscasts and in a more extensive database built from random Wikipedia articles. Specifically, inthe RTVE database, we have considered four different sets of news broadcasted: 1) *Julen* case, ten consecutive days in January 2019 when a child was trapped in a well; 2) *Notre Dame* case, ten consecutive news broadcasts in April 2019 when a fire took place at Notre Dame Cathedral (Paris, France); 3) *Singapore* case, nine consecutive broadcasts in June 2018 related to a denuclearization agreement that took place in North Korea; and 4) *Strasbourg* case, ten consecutive news broadcasts related to the terrorist attack on a Christmas market in Strasbourg, France. This database consists of one VTT file per newscast (one newscast per day), and we count on 39 VTT files about the previous topics, with differentpieces of news (different consecutive days have been considered). The main topic of each newscast corresponds to the name of the database dataset, however, there are more than 5 other news items within each newscast in the RTVE database. In the case ofthe Wikipedia database, it is an extensive dataset constructed from random articles from this popular site. We will show the process of dataset construction and the results obtained in the segmentation of the complete database. The results obtained in both datasets show an excellent performance of the proposed architecture and the novel algorithms implemented on it, improving alternative classification approaches involving algorithms such as DBSCAN [19],Agglomerative Clustering [20], K-Shifted Means [21], Mean Shift [22], and Spectral Clustering [23].

The remainder of the article has been structured in the following way: the next section describes the proposed model architecture. Section III describes two novel algorithms PBMM and FBBCM, which improve the segmentation accuracy reached by the previous modules of the architecture. Section IV evaluates the performance of our proposal in the news segmentation from newscasts of RTVE (Spanish Television) and a large dataset of random Wikipedia articles. Section V closes the article with some conclusions and remarks on the research carried out.

II. Proposed Model Architecture

The proposed architecture is depicted in Fig. 1, where 6 specific modules are defined:

1) *DatabaseManager:* this module transforms the database according to the SLM selected for the architecture. Therefore, this block reads the input subtitle file, according to the SLM chosen, and transforms it to a text file containing each sentence in the subtitle file, clearly written, in a language that the model is oriented to, compatible with the SLM. This same block must contain a set of placeholders or symbols that specify temporal jumps between sentences and other vectors containing the temporal length of each sentence.
2) *TemporalDistance Manager:* once the database is transformed, the TDM reads the file and extracts the placeholders into a placeholder vector $\mathbf{p}$, all the sentences into

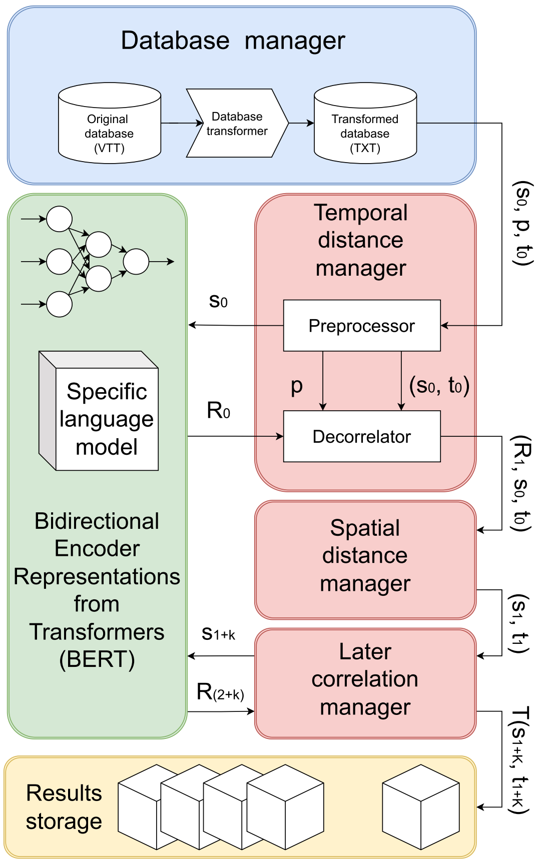

Fig. 1. Block diagram of the proposed algorithm stack.

an initial sentence vector $\mathbf{s}_0$ and time lengths into an initial temporal information vector $\mathbf{t}_0$. This module makes use of the SLM to get the correlation between each sentence in the generated $\mathbf{s}_0$ vector anduses the placeholders to modify those semantic correlation values for later modules.

3) *SpecificLanguage Model:* nowadays, complex models can perform a semantic correlation with acceptable accuracy. However, these models are not usually enough to perform news segmentation classification on their own. We use these models as a base for one of the three features taken into account for news segmentation. These models generate an embedding from a sentence, which represents the semantic information that defines it. We can get correlation values by performing mathematical transformations to this embedding and use it as information for news segmentation.

4) *SpatialDistance Manager:* about the semantic and temporal correlation values processed in earlier modules, the SDM uses this information to decide if a group of consecutive sentences belongs to the same piece of news and rearranges the initial sentence vectors $\mathbf{s}_0$ into a new vector concatenating the sentences into a new piece of text stored in the $\mathbf{s}_1$ vector. Temporal information in $\mathbf{t}_0$ may also be merged into a new $\mathbf{t}_1$ vector according toa specific proximity-based algorithm.

5) *LaterCorrelation Manager:* assuming that there is a proper algorithm that clusters consecutive sentences into a new vector of pieces of text, a separate group of sentences may correspond to the same piece of news. At this point, all temporal and spatial information should be ignored to be able to detect semantic similarities between separate pieces of text. This module is specialized in this task and makes use of the SLM, maybe iteratively, taking $\mathbf{s}_1$ as input.

6) *ResultsStorage:* once the last vectors $\mathbf{s}_{1+K}$ and $\mathbf{t}_{1+K}$ are computed, we can say each element of $\mathbf{s}_{1+K}$ is the text corresponding to each piece of news, and $\mathbf{t}_{1+K}$ is the time length of that same piece of news. The LCM module must take responsibility for transforming these sentences and timestamps into a structurethat will define the piece of news related to that element of the vector. In this architecture, we propose a Tree structure containing the embedding of the whole payload and each sentence of that piece of news, plus more useful additional information.

Further specifications of each module, such as its mathematical modeling, will be detailed in the following subsections.

A. Database Transformation

This section focuses on a pre-processing module that transforms a VTT file into a TXT file, using a specific policy. The primary objective of this pre-processing step is to enhance the accessibility of text data to the architecture and establish a universal format for processing different data types. To achieve this, the pre-processing module divides the file into individual sentences, includes temporal length information, and marks temporal discontinuities.

The pre-processing initial step of the module separates the VTT file into individual sentences and stores them in separate lines of the output TXT file. This process facilitates text data processing and analysis as each sentence can be treated as a distinct unit. The second step involves creating a vector containing the temporal length of each sentence, providing critical information for tasks such as sentiment analysis or emotion detection. The final pre-processing step introduces a token at the beginning of each new sentence if a temporal jump is detected between two sentences. The token marks temporal discontinuities in the text data, which can be useful for textsegmentation. In our proposal, the token “\$” is used to represent a time jump, as it is an exceptional symbol and does not typically appear at the beginning of sentences.

Note that the pre-processing module can handle different subtitle file formats, as long as they meet specific conditions, such as: 1) having readable chunks of text, 2) a timestamp assigned to each piece of text, and 3) the ability to split text chunks into separate sentences. To ensure a consistent format, a “Database Transformer” can be developed that modifies files to meet these conditions for each subtitle format. In addition, the proposed architecture can build several attributes from the plain text file processed by the “Database Transformer”, including a vector with each of the $\mathbf{s}_0$ phrases, a vector with the temporal length of each sentence $\mathbf{t_0}$, and a vector with the indexes of the sentences where important time jumps occurred $\mathbf{p}$. These

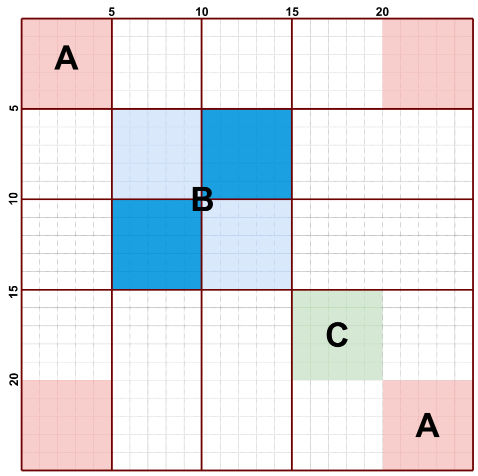

Fig. 2. Illustrative description of the problem proposed as an example.

indexes are also known as “placeholders”, and can be reconstructed by looking at the $ token for each sentence in $\mathbf{s}_0$. All the mentioned information must be reflected in the TXT file.

An illustrative example is presented to aid in comprehending the behavior of the proposed model. Let us assume an article containing a VTT file comprising 25 sentences, wherein the first five sentences pertain to news item A, the subsequent ten to news item B, the following five to news item C, and the final five to news item A once again. We have identified temporal jumps in indexes 5, 10, and 15, implying that there were narrative shifts between news items A, B, and C. These jumps are used to segment news items in the initial stage (TDM), as time jumps are common between news items but relatively infrequent within them. The DBM module extracts the temporal jumps $\mathbf{p}$ (indices 5, 10, and 15) and sentence vector $\mathbf{s}_0$. The SLM obtains the vector representation of these sentences using a pre-trained model and generates a correlation matrix based on semantic information. The TDM uses the time jump information to strategically reduce the correlation matrix values, making it simpler for subsequent modules to segment news items A and B as distinct blocks (index 5), such as B and C (index 15). However, some temporal information may not fit the topic difference and may hinder segmentation (index 10). It will be the task of future modules to overcome this drawback. The SDM employs two algorithms (GPA and PBMM) to segment the correlation matrix and combine sentences into text blocks based on the segmentation, identifying four segments (news items A, B, C, and news item A again). The SLM is applied once again to each text block to obtain a correlation matrix, as the context of the sentences changes when they are combined into a block. Since BERT uses context information, recursive calls to the SLM are necessary to update the context information. Finally, the LCM module uses the FB-BCM algorithm to identify clusters between different segments that were spatially separated but belong to the same news item, identifying news item A in two segments (the first and the last one) and grouping them. Additionally, the algorithm in the LCM module can correct errors made by previous modules during the segmentation task.

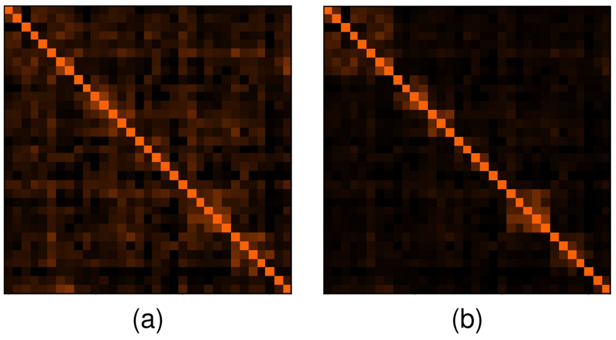

Fig. 3. Example of an $\mathbf{R}_0$ matrix obtained as an output of the specific language model module (3a) and the subsequent $\mathbf{R}_1$ matrix obtained from applying a temporal penalty factor $\beta = 0.6$ (3b).

Fig. 2 serves as an illustrative example of the proposed problem. It represents the collection of news items represented by individual squares. Each square is colored according to its corresponding news item, with red denoting news item A, blue denoting news item B, and green denoting news item C. Furthermore, temporal jumps are identified within the figure, with the initial and final temporal jumps serving as markers for the boundaries between news items. However, as previously mentioned, the middle temporal jump has a negative impact on the segmentation, leading to a perception that block B is divided into two news items.

B. Correlation Managing

The most important feature taken into account in the model is *sentence similarity*. As previously mentioned, the model responsible for this task is an open-source Natural Language Processing module [25] with a BERT [9] (BETO [17] in this case) architecture. Such functionality can be modeled as a functionthat takes a vector of sentence structures $\langle \mathbb{S} \rangle^M$ and returns a correlation matrix associated with them:

$$\begin{aligned} f_B : \langle \mathbb{S} \rangle^M &\rightarrow \mathbb{R}^{M \times M} \\ \mathbf{s} &\rightarrow \mathbf{A} = f_B(\mathbf{s}) \end{aligned} \tag{1}$$

All the components of the $\mathbf{A}$ matrix referring to (1) are correlation values included in the interval $[0, 1]$, where 0 means no correlation at all, and 1 means absolute correlation. Note that this matrix must be symmetric, and each element on the diagonal must be equal to 1, as the correlation between a sentence and the very same sentence is always maximum and 1. Fig. 3(a) represents, on an image, an example of a $\mathbf{R}_0$ correlation matrix.

BERT computes a D-dimensional vector representation for each sentence in the vector of sentence structures, in the model used [25], it computes a 768-dimensional vector ($D = 768$) from that structured vector. This set of vectors is called *embedding*.

$$\begin{aligned} f_{BERT} : \langle \mathbb{S} \rangle^M &\rightarrow \mathbb{R}^{M \times D} \\ \mathbf{s} &\rightarrow \mathbf{E} = f_{BERT}(\mathbf{s}) \end{aligned} \tag{2}$$

From these embedding, we compute the cosine similarity (cs) to obtain the correlation between all possible embedding combinations as follows:

$$cs(\mathbf{a}, \mathbf{b}) = \frac{\mathbf{a} \cdot \mathbf{b}}{\|\mathbf{a}\|\|\mathbf{b}\|} = \frac{\sum_{i=1}^{n} a_i b_i}{\sqrt{(\sum_{i=1}^{n} a_i^2)(\sum_{i=1}^{n} b_i^2)}} \tag{3}$$

After cosine similarity is computed, some values can be slightly lower than 0; this is solved by applying a negative-values-to-0 threshold in the matrix. Finally,these correlation values can be arranged into a matrix:

$$\begin{aligned} f_{CS} : \mathbb{R}^{M \times D} &\to \mathbb{R}^{M \times M} \\ \mathbf{E} &\to \mathbf{A} = f_{CS}(\mathbf{E}) \end{aligned} \tag{4}$$

This last matrix will be the corresponding output for each set of sentences, and its values will be the base of all future modifications in the model.

C. Temporal Distance Managing

Any VTT file must contain temporal information, as it is a main requisite for all Video Subtitling Software, so it is possible to print with correct timing each piece of text. This temporal information can be also used to improve newscast segmentation performance. As a recommendation, database transformation shall be performed before data processing. As VTT segments are randomly, or not optimally, distributed. In this model, we assume that a sentence is the minimum size element, considering that, each VTT segment will be arranged into sentences $(\mathbf{s}_0)$ and a temporal information vector $(\mathbf{t}_0)$. While transforming the VTT file intoa TXT file, temporal jumps over 1 s are stored into a *placeholder vector* $\mathbf{p}$. As the model works with correlation matrices, the time managing module waits for BERT to perform the transformation $\mathbf{R}_0 = f_B(\mathbf{s}_0)$ relative to the temporal information. This transformation takes as input a correlation matrix, $\mathbf{R}_0$, (real numbers), the vector with the positions of the time jumps, $\mathbf{p}$, (which is a vector of natural numbers) and a hyperparameter, β, which is a real number. A decorrelation is appliedto the input matrix with the information contained in the vector of time jumps (placeholders), using the hyperparameter to carry out the decorrelation to a greater or lesser extent. The output matrix of the decorrelator is defined as $\mathbf{R}_1$. Decorrelation is a general term for any process that can be used to reduce the degree of autocorrelation of a signal, or cross-correlation between different signals in a group given specific information. A decorrelator isa module that produces a decorrelation of a signal or correlation matrix.

Once the first correlation matrix $\mathbf{R}_0$ is obtained, a transformation f_D is computed. This transformation can be mathematically modeled as follows:

$$\begin{aligned} f_D : \mathbb{R}^{M \times M} \times \mathbb{N}_0^N \times \mathbb{R} &\to \mathbb{R}^{M \times M} \\ (\mathbf{A}, \mathbf{p}, \beta) &\to \mathbf{B} = f_D(\mathbf{A}, \mathbf{p}, \beta) \end{aligned} \tag{5}$$

where the *temporal penalty factor* β is the hyperparameterthat rules the module and represents the reduction applied in every correlation value through $\mathbf{R}_0$. This transformation creates a virtual range of intervals following (6).

$$\mathbf{I} = ([0, p_0 - 1], [p_0, p_1 - 1], [\ldots], [p_{z-1}, p_z - 1]) \tag{6}$$

Finally, the output correlation matrix $\mathbf{R}_1$ can be obtained as follows, where i is the row of the matrix, j is the column, and (i, j) is the value in row i column j:

$$R_{1_{i,j}} = \begin{cases} R_{0_{i,j}} & \text{, if } i \in I_v \wedge j \in I_v \\ R_{0_{i,j}}(1-\beta) & \text{, if } i \in I_v \wedge j \notin I_v \end{cases} \tag{7}$$

In Fig. 3(b) we can see $\mathbf{R}_1$ after a temporal penalty factor of $\beta = 0.6$, which is obtained from the initial $\mathbf{R}_0$ matrix in Fig. 3(a) provided as an example.

This matrix $\mathbf{R}_1$, is the input of the spatial managing module, that applies a spatial distance-based algorithm.

D. Spatial Distance Managing

In this module, a vector of sentence structures is taken as input with a corresponding correlation matrix $\mathbf{R}_1$ given some temporal information about the text. Here, we propose an algorithm that takes as a hypothesis that exists any kind of spatial correlation between sentences. They are clustered if they are close to each other.Despite proposing an algorithm for this module, any custom algorithm can be implemented as long as it preserves its functional structure as (8) shows:

$$\begin{aligned} f_{SP} : \mathbb{R}^{M \times M} \times \langle \mathbb{S} \rangle^M \times \mathbb{R}^M &\to \langle \mathbb{S} \rangle^N \times \mathbb{R}^N \\ (\mathbf{B}, \mathbf{s}, \mathbf{t}) &\to (\mathbf{s}', \mathbf{t}') = f_{SP}(\mathbf{B}, \mathbf{s}, \mathbf{t}) \end{aligned} \tag{8}$$

This module takes a correlation matrix, a sentence structure vector, and a temporal information vector as input and returns a new structure of a concatenated text and temporal information as output. Notice that output size *N* must be lower or equal to input size *M*.

The module uses $\mathbf{R}_1$ matrix as a reference to concatenate elements of $\mathbf{s}_0$ and add elements of $\mathbf{t}_0$ vectors. The algorithm implemented should use $\mathbf{R}_1$ information, and decide if a subset of elements in $\mathbf{s}_0$ may be concatenated into a new element in $\mathbf{s}_1$ and, in parallel, a subset of elements in $\mathbf{t}_0$ added into $\mathbf{t}_1$ according to the same directive as $\mathbf{s}_1$ vector. Notice that the text is not modified, but rearranged into text vectors according to the algorithms. This is important, as BERT models output different embeddings if the text contains one sentence or contains several sentences inside. This is the reason why sentences in $\mathbf{s}_0$ are collapsed into new text vectors in $\mathbf{s}_1$, as the next module will use this new vector to generate a new embedding for each element. In regards to $\mathbf{t}$ vectors, if several sentences are rearranged into a new piece of text into $\mathbf{s}_1$, its time length is added into $\mathbf{t}_1$ vector, now $\mathbf{t}_1$ elements represent the time length of the pieces of text in $\mathbf{s}_1$ as $\mathbf{t}_0$ did to $\mathbf{s}_0$ before. This new information is transferred to the next module.

E. Later Correlation Managing

At this point, sentences that are spatially and temporally close to each other and share semantic correlation should have been clustered. However, some blocks of sentences that correspond to the same topic can be spatially and temporally distant and

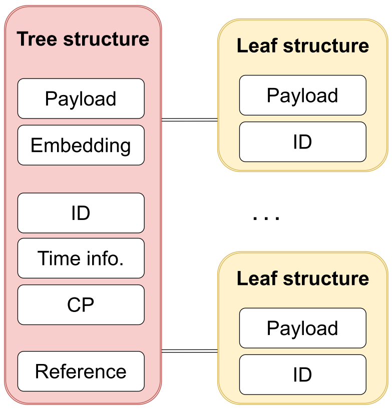

Fig. 4. Proposed structure for storage results.

should be clustered into the same block. This module solves that exact problem, as the previous modules took spatial and temporal distance as a penalty feature,but this does not. Additionally, this module can help improve earlier modules' performance.

As the model initially did (although now there is no placeholder vector p), this block takes as an input a vector of structured sentences $\mathbf{s}_1$, and a time information vector $\mathbf{t}_1$. This block can make iterative uses of f_B (Equation1) transformation, returning a $\mathbf{s}_{1+k}$ and $\mathbf{t}_{1+k}$ vectors, where k is the number of the current iteration in the algorithm, which will finish in a finite number (K) of iterations. As a functional module, this shall end up performing the following transformation:

$$f_{CM} : \langle\mathbb{S}\rangle^N \times \mathbb{R}^N \to \langle\mathbb{S}\rangle^L \times \mathbb{R}^L$$
$$(\mathbf{s}', \mathbf{t}') \to (\mathbf{s}'', \mathbf{t}'') = f_{CM}(\mathbf{s}', \mathbf{t}') \tag{9}$$

The modulecan find strong semantic correlations between separate and not separate elements of the current vector $\mathbf{s}_{1+k}$. The main reason this algorithm is likely to be iterative lies in the fact that f_B transformation can vary greatly for each k iteration, as context information used by the SLM changes every time an element of $\mathbf{s}_{1+k}$ is modified. An algorithm can be able to find strong correlations with small changes in any element. As additional work, this module will transform all the information (temporal, text, and optionally embedding) into a custom structure to save it as a result.

F. Results Storage Structure

As we propose in this study, a piece of news is not only defined by its text but it is also defined by its temporal information. Nevertheless, results storage strongly depends on the application which will make use of the model. Hence, we propose a structurethat contains the elements shown in Fig. 4. This tree transformation can be modeled as a not pointed set as (10) shows:

$$T : \langle\mathbb{S}\rangle^L \times \mathbb{R}^L \to \{\langle\mathbb{S}\rangle, \mathbb{R}^D, \mathbb{N}, \mathbb{R}, \mathbb{R}, \mathbb{N}, \langle\mathbb{S}\rangle^J, \mathbb{N}^J\}^L$$
$$(\mathbf{s}'', \mathbf{t}'') \to T_{ree} = T(\mathbf{s}'', \mathbf{t}'') \tag{10}$$

where L is the number of news classified by the model and J is the number of *leaves* or differentiated pieces of text in the tree, which depends on the decisions of the algorithm. Here, a tree structure defines a piece of news, where each element of the structure is defined as follows:

- *Payload:* defines the whole text of the piece of news, it involves all sentences related to the same piece of news combined into a single piece of text. It can be defined as a text structure $\langle\mathbb{S}\rangle$.
- *Embedding:* it is a vector of real numbersthat define a semantic representation of the payload. In this model, it is the output of the function defined in (2), the output of the SLM. It can be defined as a D dimensional vector of real numbers $\mathbb{R}^D$. This embedding is stored for computational efficiency reasons, as some models may take a long time to compute.
- *ID:* it is a natural number defining the tree identity, this number must be unique for each tree in the results storage. It can be defined as a natural number $\mathbb{N}$.
- *Time information:* it is a vector containing the whole temporal length of the piece of news. It can be defined as a real positive number $\mathbb{R}^+$.
- *Correlation power (CP):* it is a real number indicating how correlated the sentences of the leaves are within the tree. This number can become very interesting when studying the reliability of algorithms. It can be defined as a real positive number $\mathbb{R}^+$. Correlation power is defined in (11):

$$p_t = \frac{2}{M\,(M-1)} \sum_{x=0}^{M-2} \sum_{y=x}^{M-1} |R_{x,y}|^2 \tag{11}$$

where M is the size of $\mathbf{R}_{1+K}$ and $\mathbf{R}$ is, in our architecture, the very last output matrix $\mathbf{R}_{1+K}$.
This function does not take into account the main diagonal of the correlation matrix as it does not provide any information about the correlation between sentences. The correlation power is defined on the $[0, 1]$ interval, meaning 0 is no correlation between any sentence in the tree, and 1means absolute correlation between all the sentences within the tree. This measurement helps to evaluate the reliability of the model. Equation (11) also appears with a similar form in other recent algorithms [11], as it is very similar to typical signal processing measurements.
- *Reference:* when several trees share the same results storage system, it is convenient to define a group of trees thatrefer to a group. For example, if analysis for several days where some piece of news can be repeated and those trees are later merged into a subsequent tree, it is convenient to reference the day those trees belong to. It can be done by its reference field and it can be defined as a natural number $\mathbb{N}$.
- *Leaves:* this structure stores information about the initial state of the model. Each leaf stores a unique *ID* value and a *Payload* value containing the minimum text size element considered; in this architecture, this element is a sentence, but a single word or any group of words could be also considered.

These tree structures contain enough information about the piece of news for later usage and it requires no further processing beyond file writing and reading.

III. Proposed Algorithms

To complete the proposed architecture, we propose two different algorithms which implement the functionalities of each layer. We propose an algorithm for the SDM module (PBMM) and another one for the LCM module (FBBCM). The first algorithm assumes as an initial hypothesis that a spatial correlation among pieces of text exists, and both focus on clustering consecutive pieces of text, based on the input matrix $\mathbf{R}_1$. This type of algorithmtries to find borders in the matrix to cluster these blocks. The second algorithm ignores all spatial and temporal distances and focuses on clusteringhighly correlated blocks.

A. Spatial-Based Algorithm: PBMM

The first proposed algorithm for the SDM module is the PBMM. This algorithm contains three differentiated steps: *mean correlation clustering*, *one in the middle*, and *check back*. As a first step, the algorithm initializes a new block. A block is a container for elements in the input matrix $\mathbf{R}_1$. Initially, the algorithm adds the first element of the matrix into the first block, and every step checks if this new element is correlated with each element in the same block, this correlation is computed as (12) shows, which perform an average of each correlation value in the matrix, where v is the index of the last element in the previous block or equals -1 ($v = -1$) if there is no previous block:

$$\mu_i = \frac{1}{i - v - 1} \sum_{j=v+1}^{i-1} R_{1_{i,j}} \tag{12}$$

note that when computing the average correlation between elements, at least two elements must be considered as (12) has an indeterminacy when $i = v + 1$.

Once computed the mean correlation for this new element in the blockis, a threshold is applied (*PBMM-th*) which will be the first parameter of the algorithm. If this value is greater than the *PBMM-th*, the algorithm considers that a correlation between the block and the element exists and adds the next element to the block. This operation is performed until a mean value for any new element is lower than the *PBMM-th*, where the algorithm considers no correlation between that element and the block. Here, there is a second chance, the algorithm ignores this fault and carries on adding the next element to the block. When several consecutive elements do not correlate with the current block, they are extracted from the current block and create a new block for them. The number of consecutive non-correlated elements, until the block is finished, is the second parameter of the algorithm (*OIM*). This parameter may be adapted according to the noise in the database, as this second chance to find new correlated consecutive elements can be accurate when some sentences are incorrectly formed or there are random intrusive phrases. For instance, when a speech recognition module creates text from a news broadcast, this module can misunderstand some words, so the embedding obtained from the SLM corresponding to this phrase is not correct and does not match the semantics of the block corresponding. Also, in news broadcasts, random intrusive phrases such as "*good night*" can appear, which do not match the semantics of the corresponding block. That is why this second chance is introduced.

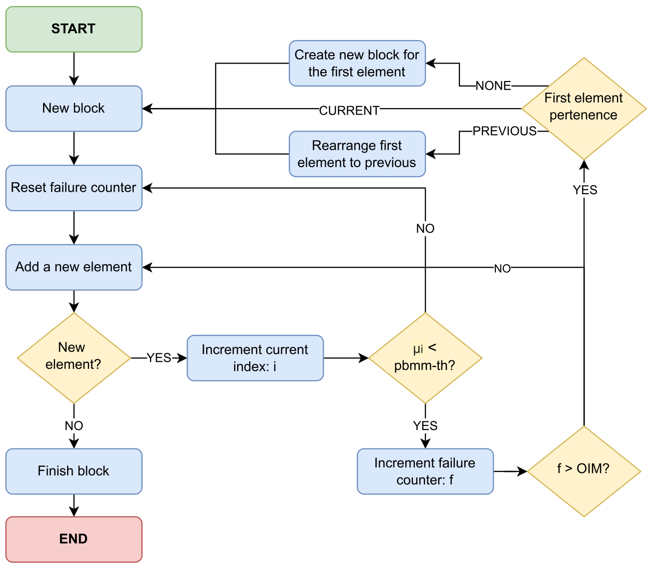

Fig. 5. Three-step Proximity-Based Merger with Means (PBMM) algorithm for spatial distance clustering.

An additional last step is introduced in the algorithm. Only in the borders between two blocks, an incorrectly formed sentence can contain semantic information of two consecutive blocks. This happens because some speech recognition modules do not enter punctuation marks correctly in some cases, so one sentence can also contain information between two different pieces of news. As it is correct to classify this corrupted sentence to any of these blocks, it is recommended to join this same sentence to the most correlated block, even if its mean value is greater than the *PBMM-th*. This step (*checkback*)checks out whether the last element of the block corresponds to the next block orto the previous one, improving slightly the algorithm performance. We use a different threshold value (*checkback threshold*) to evaluate if the borderline element corresponds to the same block orto the previous block. The PBMM algorithm is presented as a flowchart in Fig. 5.

For some specific algorithms, a proximity adapter can take place to improve the overall module performance. In the next subsection, we propose a specific proximity adapter to implement. This adapter modifies the original input matrix based on the spatial distance. In Fig. 6 the overall scheme of a complete SDM is presented. An initial process for the adapter is taken place in the SDM. After the first sub-module, the main algorithm takes place and determines the segmentation $\mathbf{d}$ for each element in the matrix $\mathbf{R}_1$. After determining which elements in the sentence vector $\mathbf{s}_0$ and temporal information $\mathbf{t}_0$ belong to the same block, they are merged into the same piece of text, and arranged into the vectors $\mathbf{s}_1$, and $\mathbf{t}_1$ following the $\mathbf{d}$ markers.

B. Spatially Based Algorithms: GPA Pre-Module

A spatial distance adapter can be implemented to improve the performance of some spatial-based algorithms. This penalty

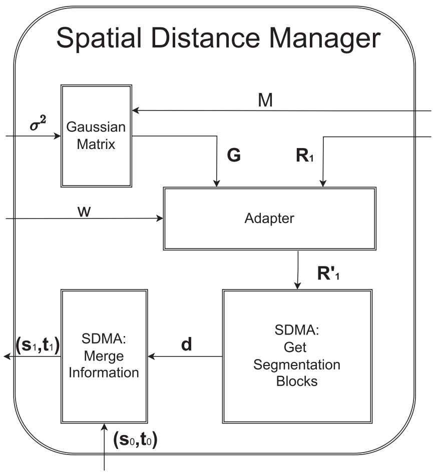

Fig. 6. Spatial Distance Managing module with GPA sub-module and Spatial Distance Manager Algorithm module.

function encourages correlation values close to each other to be greater than those that are far apart. Several functions can be implemented for this purpose. Here, we propose to use either an exponential function [2] or, better, a Gaussian Proximity Adapter (GPA). It should be noted that this moduledoes not always helps improve the overall performance, as, after this adapter, it is harder for the SDM algorithm to distinguish between borderline elements.Among the proposed stack of algorithms, it is not implemented due to this issue, but it should be considered for other custom algorithms.

$$f_{GPA} : \mathbb{R}^{M\times M} \times \mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R}^{M\times M}$$
$$\mathbf{R}_1 \rightarrow \mathbf{R}'_1 = f_{GPA}(\mathbf{R}_1, \sigma^2, w) \quad (13)$$

This sub-module generates a GPA matrix to combine with the original input matrix $\mathbf{R}_1$. The GPA module takes 3 inputs: *variance of the Gaussian* σ^2, the size of the original matrix, and the weight w of the module. This weight represents the importance we want to give to the adapter (in the interval $[0, 1)$), taken as a parameter. The variance provided strongly depends on the database, for short pieces of news it will work better with low values of variance, while for larger pieces of news, it will work better with higher values. The output matrix can be obtained as follows:

$$\mathbf{R}'_1 = (1 - w)\mathbf{R}_1 + w\mathbf{G} \quad (14)$$

where $\mathbf{G}$ is the GPA matrix and w is the weight of the sub-module. The GPA matrix can be obtained as (15) shows:

$$G_{i,j} = e^{-\frac{(i-j)^2}{2\sigma^2}} \quad (15)$$

When using the PBMM algorithm, this sub-module can be skipped (as it reduces the algorithm performance) or introduce a weight of 0 ($w = 0$) as a parameter value to nullify the effect of the GPA matrix and use a raw $\mathbf{R}_1$ correlation matrix.

TABLE I
OPTIMAL PARAMETER COMBINATION FOR THE VALIDATION DATABASE

TDM	SDM				LCM	SLM
β	w	OIM	pbmm-th	cb-th	fbbcm-th	Model
0.25	0	1	0.177	0.147	0.614	BERT [25]

C. Later Correlation Algorithm: FBBCM

The second proposed algorithm is FBBCM which stands forFavorite-Based Bidirectional Connection Merging. This algorithm tries to find a strong semantic correlation between pieces of text even if they are spatially and temporally distant. Given a set of pieces of text inside the vector $\mathbf{s}_1$, this algorithm makes use of (1) transformation to get new embedding from each piece of text and obtain a new correlation matrix between those pieces of text. Once the algorithm considers that two pieces of text are strongly correlated, they are merged into the same piece of text and stored into an element of the vector $\mathbf{s}_2$ for the next iteration, preserving their occurrence order in the text. In the next iteration, the same procedure is done with $\mathbf{s}_2$ vector as input, building a new vector, $\mathbf{s}_3$. Each iteration of the algorithm produces a new correlation matrix of the current $\mathbf{s}_{k+1}$ vector where k is the number of the iteration which is taking place algorithm. This is so because the same pieces of text concatenated produce a different embedding and all correlation values can change, as BERT models also make use of contextual information. When the algorithm considers that there is no more strong correlation between pieces of text, the algorithm ends, taking a total number K of iterations. The algorithm takes a look into the current $\mathbf{R}_{k+1}$ matrix and checks the highest value for each row, the index of the highest value is named its*favorite* block, if two blocks are *bidirectionally favorite*, which means that one general block α^1 has its maximum correlation value for block α^2, block α^2 has its maximum correlation value for block α^1 and both correlation values are greater than the threshold of the algorithm (*FBBCM-th*), they are considered strongly correlated and merged for the next iteration. This threshold (*FBBCM-th*) is introduced to avoid divergence and, generally, should be greater than the previous algorithm threshold (*PBMM-th*). Algorithm 1 shows the FBBCM algorithm for the LCM module.

IV. Performance Evaluation

Different experiments have been carried out to test the performance of the proposed architecture in real SSBN problems. Aspreviously mentioned, experiments in RTVE and Wikipedia datasets have been considered.First,the RTVE database has been used to obtain optimal parameter values for each module, as Table I shows. Specifically, this optimal parameter adjustment has been carried out for *Julen* news broadcast database, assigned for validation. This news corresponds to a set of 10 Spanish news broadcasts related to a trapped child in a well, in January 2019. For testing, 3 more databases have been taken into account: *Notre Dame*, related to the fire at Notre Dame Cathedral (Paris, France) in April 2019 (10 news broadcasts); *Singapore*, related to an agreement on the denuclearization of North Korea in June

Algorithm 1: FBBCM Algorithm for the Later Correlation Manager (LCM) Module.

```
Input: Text vector $\mathbf{s}_1$, temporal information $\mathbf{t}_1$, threshold, $f_B$.
Output: Merged text vector $\mathbf{s}_{1+K}$, merged temporal information $\mathbf{t}_{1+K}$.
Iterative Process
 1: changed = true
 2: k = 0
 3: while changed do
 4:     $\mathbf{R}_{2+k} = f_B(\mathbf{s}_{1+k})$
 5:     Find the index of the maximum value for each row in $\mathbf{R}_{2+k}$ if it is greater than FBBCM-th and store it into I.
 6:     Find all the pairs of indexes which meet the conditions: I(i1) = i2 and i2 > i1.
 7:     $\mathbf{s}_{2+k}$ = merge($\mathbf{s}_{1+k,i1}, \mathbf{s}_{1+k,i2}$) for each pair $(i1, i2)$ of matched indexes.
 8:     $\mathbf{t}_{2+k}$ = merge($\mathbf{t}_{1+k,i1}, \mathbf{t}_{1+k,i2}$) for each pair $(i1, i2)$ of matched indexes.
 9:     if $(s_{1+k} = s_{2+k})$ then
10:         changed = false
11:     end if
12:     k = k + 1
13: end while
14: return $\mathbf{s}_{1+k}$, $\mathbf{t}_{1+k}$
```

2018 (9 news broadcast); and *Strasbourg*, related to a terrorist attack on a Christmas market in Strasbourg, France, in December 2018 (10 news broadcast).

Additionally, due to the reduced length of the database focused on the specific problem of broadcasted news in RTVE, we have performed several experiments on a much larger database. This database uses Wikipedia [24] to generate a database with random articles. These articles do not contain temporal information, so it is a database that only contemplates text segmentation by subject and not subtitles. With this database, we will demonstrate that this architecture is also efficient for text segmentation and not only for news. However, temporal information can be modeled with a probabilistic approach and can also be used to model informative texts such as news.

The performance metrics used in this analysis are F1 score, Precision (P), Recall (R), WindowDiff (WD), and Pk score (Pk) which are widely-used metrics for story segmentation (WD and Pk) and classification (P, R, F1). Higher values of F1 score and high balanced values for Precision and Recall are indicators of better classification performance, while higher values for WindowDiff or Pk score indicate a worse segmentation performance, as it evaluates an error. F1 is related to Precision and Recall as the harmonic mean. For evaluation, all words were taken into account for each piece of news, and no tolerance was considered. This analysis considers the number of words classified (or not classified) into the correct piece of news, even if the database contains noise and it is not possible to cluster some specific words properly. Note that for segmentation modules (TDM and SDM) the main metric will be WindowDiff (WD), while the classification module (LCM) will consider the F1 score instead, especially for optimization tasks.

Regarding the trending evaluation metrics for text segmentation, as recommended in [26], WindowDiff is shown in (16) and $\mathbf{P_k}$**-score** in (17). These methods use a sliding window to calculate the metric. Note that all the metrics for text segmentation are error metrics, and the smaller their value the better the performance. They are similar equations, and their values are bounded in the range [0,1]. The segmentation boundaries are binary vectors where 0 represents a non-boundary element and 1 represents a new segment in the current element. This is a common codification for segmentation tasks. These metrics have as input the hypothetical segmentation (hyp), thereference segmentation (ref), and a parameter that defines the window size (k). Both use specific functions p (P_k) and b (WD).Here b is a function that counts the number of segmentation boundaries in the first given parameter; between the indexes given in the other two parameters.p is a function that determines the existence of boundaries between the indexes provided as inputs. The main difference between p and b is that p returns 1 if a boundary exists and 0 if not, while b returns a natural number lower than the window size k:

$$WD(hyp, ref, k) = \frac{1}{N-k}\sum_{i=1}^{N-k}(|b(ref, i, i+k) - b(hyp, i, i+k)|) \tag{16}$$

$$P_k(hyp, ref, k) = \frac{1}{N-k}\sum_{i=1}^{N-k}(|p(ref, i, i+k) - p(hyp, i, i+k)|) \tag{17}$$

where, k is the window size; N is the number of elements; hyp is the hypothesis segmentation, in our case, the hpy; ref is the reference or Ground Truth segmentation; p is the segmentation membership function; b is the number of boundaries function; P_k is the P_k-score; and, WD is the WindowDiff-score. The window size k has been adjusted as recommended in [27]:

$$k = \left[\frac{L}{2(1+N_B)}\right], \tag{18}$$

where L is the number of sentences (common for *ground truth* and performed) and N_B is the number of boundaries in the *ground truth* segmentation. Note that $[\cdot]$ represents the nearest integer to the argument.

A. Experiments in RTVE Newscasts

Table II shows the importance of each module inside the architecture as a first comparison inthe RTVE database. The performance of the architecture has been measured for each combination of active and non-active modules on it: the temporal correlation manager (TCM), spatial correlation manager (SCM), and later correlation manager (LCM). Table II confirms that the best combination of modules is when all of them are active. Even though precision can be higher in some cases, the best

TABLE II
MODEL PERFORMANCE BASED ON F1 SCORE, PRECISION, RECALL, AND WINDOWDIFF DEPENDING ON THE ACTIVATION OF THE DIFFERENT MODULES, FOR THE RTVE DATABASE

Active modules			Julen (validation)				Notre Dame				Singapore				Strasbourg			
LCM	SDM	TDM	P	R	F1	WD	P	R	F1	WD	P	R	F1	WD	P	R	F1	WD
No	No	No	**1.00**	0.58	0.73	0.99	**1.00**	0.59	0.74	0.99	**1.00**	0.59	0.73	1.00	**1.00**	0.62	0.76	0.99
No	No	Yes	**1.00**	0.58	0.73	0.99	**1.00**	0.59	0.74	0.99	**1.00**	0.58	0.74	1.00	**1.00**	0.62	0.76	0.99
No	Yes	No	0.76	0.84	0.79	0.38	0.74	0.85	0.79	0.33	0.73	0.74	0.72	**0.38**	0.71	0.83	0.75	0.42
No	Yes	Yes	0.87	0.83	0.84	0.37	0.86	0.81	0.83	0.34	0.82	0.74	0.77	**0.38**	0.78	0.81	0.78	**0.41**
Yes	No	No	**1.00**	0.59	0.74	0.98	0.99	0.60	0.75	0.98	0.98	0.61	0.75	0.96	**1.00**	0.63	0.77	0.96
Yes	No	Yes	**1.00**	0.59	0.74	0.98	0.99	0.60	0.75	0.98	0.98	0.61	0.75	0.96	**1.00**	0.63	0.77	0.96
Yes	Yes	No	0.72	**0.90**	0.79	0.38	0.74	**0.89**	0.80	**0.32**	0.70	**0.87**	0.77	**0.38**	0.70	**0.87**	0.76	0.42
Yes	**Yes**	**Yes**	0.87	0.89	**0.88**	**0.33**	0.85	0.84	**0.84**	0.33	0.76	**0.87**	**0.80**	**0.38**	0.78	0.85	**0.80**	**0.41**

The bold values highlight the best results for each column (metric).

F1 score is achieved when every module is active. When we refer to activating or deactivating modules, it means configuring their hyperparameters in such a way that they do not modify the information passing through them (to deactivate the module), and or set them to their optimal values (to activate the module). For instance, in the TDM module, we set $\beta = 0$ to deactivate it, so there will be no modification of the correlation matrix coming from the SLM, whatever the placeholder vector $\mathbf{p}$ is. For the SDM, we set the deactivation parameters to $pbmm\_th = 1$, so that there is no segmentation and everything corresponds to the same block. Finally, for LCM we set $fbbcm\_th = 1$ so that there is no post-segmentation clustering task.

Precision is high when SDM is not active, as a result of not clustering any sentences. Every sentence is classified as a different piece of news, so there is no failure in guessing. Despite that, recall is low as there is no clustering. Note that using semantic similarity is always required, as it is the basis for all the modules proposed. In addition, when LCM is not active, segmentation performance (WD) is very close to when it is active, as LCM only classifies segmented blocks, which may change a few segmentation boundaries but it is not the main task of the module. The best segmentation performance is achieved when both TDM and SDM are active. Another notable result is that when SDM is not active, the architecture performance greatly decreases, as only LCM segmentation is deficient. TDM increases the performance by 4-5% in the test dataset and 10% in the validation dataset. We emphasize that LCM is necessary to improve the performance of the model. Table II shows a section where we disable the LCM and we can see that the Recall is lower. This makes sense since the LCM is dedicated to grouping segments if they belong to the same piece of news. We can also analyze these results in terms of error propagation in the system. Note that accurately determining the impact of recursive calls to the SLM module and breaking the sequential loop in the LCM is impractical. However, the experiment conductedin Table II reveals that this type of error propagation does indeed occur and can be estimated. The SLM has a significant influence on the model error, as evidenced by its presence as the primary factor in the table. The other errors listed in the table are attributable to the contribution of each module to the SLM error. The SDM is the second-highest contributor to the error (achieving 0% of correct boundaries if deactivated to 66% of correct boundaries if activated, in the best case), followed by the TDM (improves the F1 score from 3% to 9%), while the LCM has the smallest effect on error (improves the F1 score from 1% to 3%). Calculating the SLM error is a challenging task, as it involves closed loops in the LCM, and models like BERT, which relies on transformers, and has a high non-linear behavior. However, the SLM can not be deactivated without largely affecting the performance of the model, as it is the baseline of all the architecture.

Fig. 7 shows the model performance for variations in the different parameters for the algorithms in a second comparison.

In Fig. 7(a), (b) and (c) we can see the different performance metrics for modifications in the main parameters of the architecture (Pk and WindowDiff for segmentation (TDM and SDM), and Recall, Precision and F1), in RTVE database. The best parameters are chosen for the best performance and balance. In Fig. 7(d) we implemented the 3 models for open-source Spanish SLM (Spanish sentence similarity vector representation). The first model (Model 0) [25] is the default SLM in the architecture, it has worse segmentation performance but greater and balanced overall classification performance. The second model (Model 1) [16] achieves the best segmentation score, however, the overall classification of the news is not the greatest, and Precision-Recall is balanced. The third model (Model 2) [28] does not converge to reliable solutions, as it is a high Recall model; this means that all the sentences are classified as one new, and this architecture can not use this model as an SLM. Note also that best performance values from validation to test may vary.

Table III shows the performance of the architecture for different well-known unsupervised clustering algorithms as a last comparison in this database. FBBCM-PBMM are the base algorithms proposed for this architecture. In Table III some typical algorithms for unsupervised classification were included inside the architecture, replacing the base algorithms. Before testing each algorithm in the architecture, all parameters were optimized in the first instance for the validation dataset (Julen) and later tested on the three test datasets. For this news segmentation and classification task, the combination of the proposed algorithms reaches the highest score for almost all the performance metrics, being *Mean Shift* better in Precisionin all the test datasets, but worse F1 score. Regarding segmentation performance metrics (WindowDiff), our algorithms reach much better results than

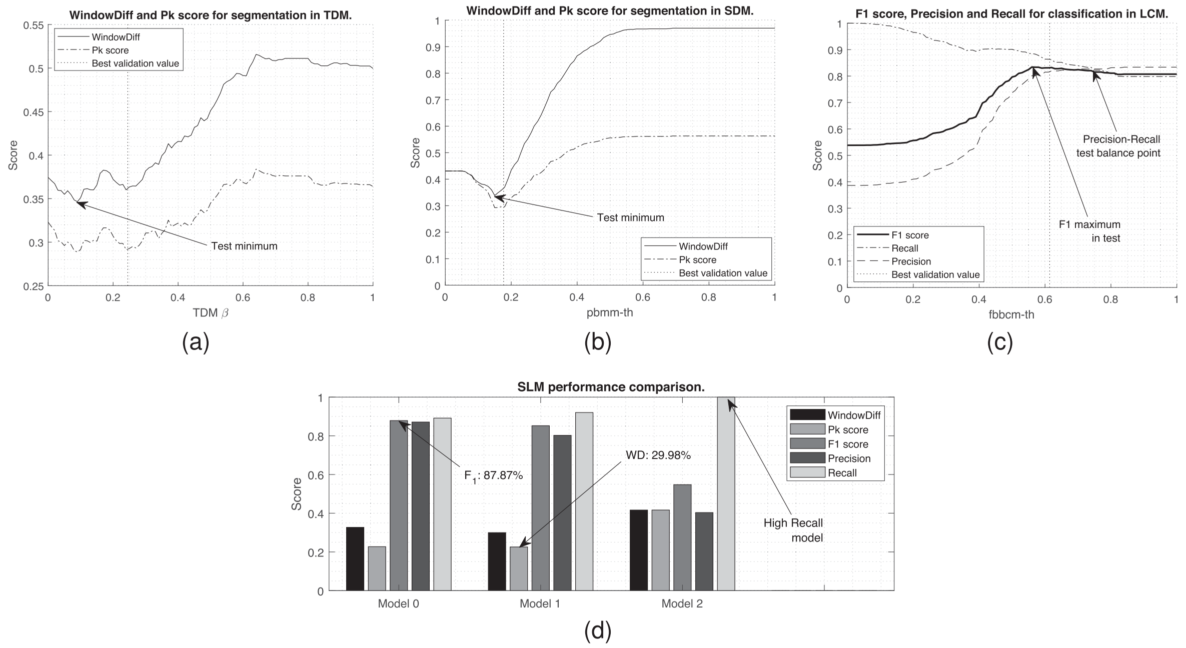

Fig. 7. F1-measure, precision, and recall for the proposed architecture and algorithms with (a) β penalty factor variation; (b) with PBMM-th variation; (c) with FBBCM-th variation, (d) specific language model variation.

TABLE III
ALGORITHM PERFORMANCE COMPARISON FOR TEXT SEGMENTATION INSIDE THE PROPOSED ARCHITECTURE

Algorithm	Julen (validation)				Notre Dame				Singapore				Strasbourg			
	P	R	F1	WD	P	R	F1	WD	P	R	F1	WD	P	R	F1	WD
PBMM-FBBCM	**0.87**	**0.89**	**0.88**	**0.33**	0.85	**0.84**	**0.84**	**0.33**	0.76	**0.87**	**0.80**	**0.38**	0.78	**0.85**	**0.80**	**0.41**
PBMM only	0.87	0.83	0.84	0.37	0.86	0.81	0.83	0.34	0.82	0.74	0.77	0.38	0.78	0.81	0.78	0.41
FBBCM only	1.00	0.59	0.74	0.98	0.99	0.60	0.75	0.98	0.98	0.61	0.75	0.96	1.00	0.63	0.77	0.96
DBSCAN [19]	0.71	0.66	0.68	0.81	0.68	0.71	0.68	0.75	0.69	0.67	0.68	0.73	0.67	0.72	0.69	0.59
K-Shifted Means [21]	0.72	0.71	0.71	0.80	0.69	0.70	0.69	0.69	0.68	0.66	0.67	0.75	0.61	0.71	0.65	0.63
Mean Shift [22]	0.86	0.71	0.77	0.82	**0.90**	0.75	0.81	0.72	**0.87**	0.69	0.76	0.86	**0.79**	0.71	0.74	0.68
Agglomerative Clustering [20]	0.71	0.69	0.70	0.78	0.74	0.75	0.74	0.64	0.70	0.66	0.68	0.72	0.66	0.75	0.70	0.52
Spectral Clustering [23]	0.76	0.66	0.70	0.87	0.76	0.68	0.71	0.76	0.72	0.62	0.66	0.82	0.69	0.68	0.69	0.67

The bold values highlight the best results for each column (metric).

other classification algorithms. For segmentation tasks, *Agglomerative Clustering* seems to behave better than others; however, our algorithms reach the highest scores for classification (F1) and segmentation (WindowDiff or Pk score) for all the datasets.

Comparing this architecture to others in the literature, this architecture performs a 36.25% WindowDiff score in noisy databases and an 83% F1 score in the same databases. Note that this architecture does not require the number of newsin advance when several of the state-of-the-art models have to know. Note also that WindowDiff score is high for noisy databases, as an out-of-phase segmentation is considered as not correct; for instance, in our databases, a "*good night*" sentence is frequent between segmentation boundaries and can be considered part of any piece of news or a different one depending on the current state of the algorithm.

B. Experiments in Wikipedia Database

In the experiments involving the Wikipedia database, the optimization performed in the RTVE problem (Table I) will be reused, as it is optimized for informative databases. While the optimization of hyperparameters can be performed for each problem that the architecture may encounter, it is unnecessary to demonstrate the efficiency of the SLM and the usability of each module in this database again. The database itself is constructed by accessing random Wikipedia articles to gather text information, which enables the creation of a database with various topics similar to news items in the RTVE database. The architecture's segmentation work (SDM) will be tested with these articles, while the clustering task (LCM) will be evaluated by separating the text into blocks, as it would be done with news. However, note that temporal information needs to be modeled since it is absent in the database. This is done by defining two probabilities, namely the probability of true detection (p_D) and the probability of false detection (p_{FD}), which correctly models RTVE newscasts as there are pauses between two news items that can be identified as large time jumps. Nevertheless, there is a possibility that this time jump is not detected (modeled with p_D) or that the jumps are detected without modeling the change in the topicsthrough the story (modeled with p_{FD}).

Algorithm 2: Database Construction Process.

```
Input: Different probabilities of detection $\mathbf{p}_D$, different
  probabilities of false detection $\mathbf{p}_{FD}$, number of newscast
  per combination $n\_files$, the statistic range of the
  maximum number of news in the article $max\_len$, the
  statistic range of the max length of the article measured in
  sentences $max\_sent$.
Output: A list of datasets for each combination in $\mathbf{p}_D$ and
  $\mathbf{p}_{FD}$.
Iterative process:
1:  $list\_of\_datasets = empty\_list()$
2:  for each combination of ($\mathbf{p}_D$ and $\mathbf{p}_{FD}$) repeated
    $n\_files$ times do
3:    $articles = empty\_list()$
4:    $overlens = empty\_list()$
5:    $n\_articles = 0$
6:    $number\_of\_news =$
      $rand\_int\_uniform(max\_len)$
7:    while $n\_articles < number\_of\_news$ do
8:      $article = read\_random\_article()$
9:      if $rand\_uniform(0,1) < p_D$ then
10:       $add\_temporal\_jump(article)$
11:     end if
12:     $ru = rand\_int\_uniform(max\_sent)$
13:     $overlen = split\_overlen(article, ru)$
14:     $n\_articles = n\_articles + 1$
15:     $articles = add\_to\_list(article)$
16:     $overlens = add\_to\_list(overlen)$
17:   end while
18:   $articles = add\_to\_list(overlen)$
19:   for $sentence$ in $article$ do
20:     if not $has\_temporal\_jump(sentence)$ and
        $rand\_uniform(0,1) < p_{FD}$ then
21:       $add\_temporal\_jump(sentence)$
22:     end if
23:   end for
24:   $list\_of\_datasets = add\_to\_list(articles)$
25: end for
26: return $list\_of\_datasets$
    where, $read\_random\_article()$ reads a random article
     ofWikipedia and split it in sentences, $empty\_list()$
     generates a empty list, $split\_overlen(article, number)$
     splits set of sentences in $article$ in two sub-sets; one
     from sentence 0 to sentence $number$; and other from
     $number + 1$ to $2 \cdot number$, and,
     $add\_temporal\_jump(sentence)$ adds the temporal
     jump in the given sentence.
```

Algorithm 2 is used to construct the database, and takes in several inputs, such as the different probabilities of detection, the number of newscasts per combination, the maximum length of the article measured in sentences, and the maximum number of news in the article. The output is a list of datasets for each combination in p_{FD} and p_{FD}. The iterative process of the database construction begins with an empty list of datasets and for each combination of (p_D and p_{FD}), repeated "n-files" times, articles are randomly selected and added to the list. If a temporal jump is detected in the article with probability p_D, it is added. The article is then split into overlapping blocks of sentences, and a temporal jump is added to each block with probability p_{FD}. The resulting articles and overlapping blocks are added to their respective lists, and a final list of datasets is returned. After running this algorithm, we will get "n-files" newscast for each combination of probabilities. We decided to select the following parameters for this experiment, as they seem to model the RTVE newscasts properly:

- $\mathbf{p}_D$: $(0.8, 0.7, 0.6, 0.5, 0.4, 0.3, 0.2, 0.1, 0.0)$
- $\mathbf{p}_{FD}$: $(0.25, 0.20, 0.15, 0.10, 0.05, 0.00)$
- n_files: 200 newscasts per combination.
- max_len: Between 4 and 9 news per article.
- max_sent: Between 10 and 14 sentences per article.

In total, there are 9 different probabilities of detection and 6 different probabilities of false detection, resulting in 54 total combinations. Therefore, the algorithm will generate a total of 10,800 synthesized newscasts (54 combinations $\times$200 newscasts per combination), which is enough for testing the architecture performance with high accuracy. The chosen range of probabilities for detection and false detection in the algorithm covers a broad spectrum of possible values, rendering it unnecessary to consider values outside these bounds. The selected range of probabilities for detection lies between 0.0 and 0.8, while the range for false detection lies between 0.0 and 0.25. This range encompasses both high and low values for these parameters, providing the algorithm with the ability to generate datasets that exhibit a diverse range of temporal structures and segmentation and clustering difficulties.

Selecting probabilities that are too extreme may produce datasets that are not informative or meaningful. For instance, selecting a probability of 1 for detection would need the introduction of a temporal jump in every article, which is not representative of real-world data. In addition, selecting a probability near 1 for false detection would encourage the introduction of too many false jumps, which again is not representative of real-world data. Therefore, the chosen range of probabilities represents a balance between providing a broad range of possible values while still generating informative datasets, as well as in the RTVE database.

Table IV shows the results of the proposed architecture inthe Wikipedia database, in terms of the mean, standard deviation, best, and worst results, as well as the median, the first quartile (Q1), the third quartile (Q3), the interquartile range (IQR), and the number of outliers for the five evaluation metrics: Precision, Recall, F1, WD (WindowDiff), and Pk-score. The results show that the proposed architecture performed well in this database with a mean F1 score of $90.30 \pm 7.76\%$ and a median F1 score of 92.15. The Precision and Recall scores had a mean of 88.54 and 92.89, respectively, indicating that the proposed model performed well in both identifying true positives and avoiding false positives. The standard deviation of the evaluation metrics was relatively low, indicating that the model's performance was consistent across the different evaluation metrics. This proves the stability of the architecture.

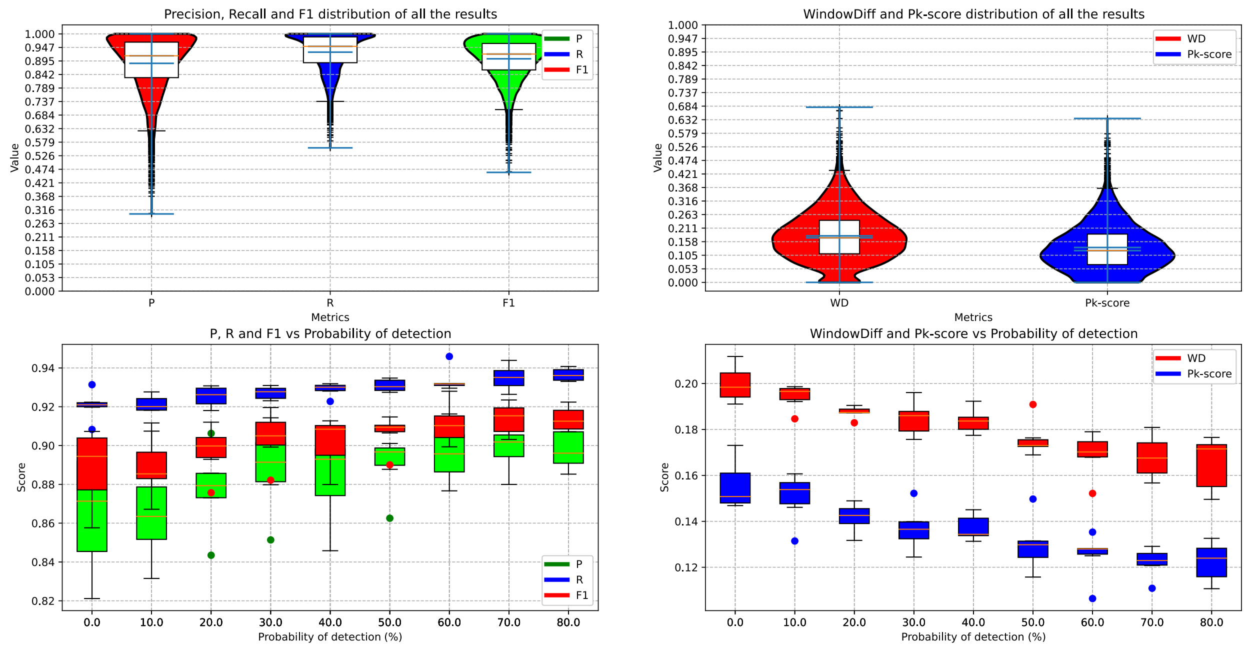

Fig. 8. Performance distribution of Precision, Recall, F1, WD, and Pk-score on the 216 blocks in the dataset depending on the p_D and p_{FD} of the temporal jumps.

TABLE IV
PRECISION, RECALL, F1, WINDOWDIFF AND PK-SCORE EVALUATED ON THE WIKIPEDIA DATABASE (%)

	Precision	Recall	F1	WD	Pk-score
Mean	88.54	92.89	**90.30**	**18.05**	13.53
Std.	10.62	07.17	07.76	09.87	08.93
Worst	30.00	55.69	**46.15**	**68.00**	63.64
Best	100.0	100.0	**100.0**	**00.00**	00.00
Med.	91.47	95.13	92.15	17.28	12.33
Q1	82.99	88.79	85.98	11.11	06.90
Q3	96.78	98.82	96.26	24.11	18.75
IQR	13.80	10.03	10.28	13.00	11.85
Outliers	02.80	01.64	02.17	01.28	01.43

The bold values highlight the best results for each column (metric).

The worst results obtained were for the WD metric, with a score of 68.00, and for the Pk-score metric, with a score of 63.64, indicating that there is room for improvement in detecting the boundaries of news segments. The first quartile (Q1) and third quartile (Q3) for all the evaluation metrics were relatively high, indicating that the model performed well in the majority of cases, and suggesting that was robust and performed consistently across most of the experiments. This is also reflected in the low number of outliers (under 3% in all cases), and in the interquartile range (IQR),which was also relatively small, indicating that the results were consistent across different segments of news. Fig. 8 shows at the top the distribution of the metrics obtained by the proposed architecture inthe Wikipedia dataset in two box and violin plots.

It is worth mentioning that these metrics have been obtained from the results of the entire database, regardless of the detection and false detection probabilities. More detailed information on the behavior of the algorithm as a function of the quality of the temporal information provided is provided below. For this purpose, we have divided the results into 4 sub-blocks within each combination of probabilities (blocks of 50 on datasets of 200 newscasts), and we have computed the average of these blocks to work with more compact results. So we have 4 performance averages for each combination of probabilities (216 in total). This performance distribution is shown in Fig. 8, where Precision, Recall, F1 score, WD, and Pk-score are shown for each of the detection, and false detection probabilities. The results obtained show, as expected, that when we increase the probability of detection, the performance of the proposed model improves. It affects Precision more than Recall, but both metrics increase along with the probability of detection. The TDM module is a support module for subsequent segmentation, so the segmentation task, evaluated with WD and Pk-score, also improves when the detection probability increases. As a general rule, Precision is below Recall, so the model is incorrectly identifying the segmentation because of TDM behavior due to the lack of information about the temporal jumps when the p_D is low. Whether Recall is good or bad depends largely on the LCM module, which does not directly use temporal-jumps information to classify. However, since segmentation error affects classification, the worse this segmentation is, the worse Recall will be.

C. Qualitative Comparison With State-of-The-Art Algorithms

Machine Learning, Deep Learning, and Statistical models applied to story segmentation usually need the number of clusters to work properly [13], [14], [15]. This is an issue for the flexibility of the models based on supervised approaches. Another flexibility issue is related to the number of data required for multi-parameter models, which is not suffered by unsupervised

TABLE V
QUALITATIVE COMPARISON BETWEEN THE STATE-OF-THE-ART AND THE PROPOSED MODEL

Algorithm	Performance	Number of parameters	Assumptions	Data dependence	Overall flexibility
Proposed architecture	F1 around 90%	Less than 10	**Temporal information**	**Very Low**	**Excellent**
Deep Learning models	**F1 around +90%**	Greater than 100,000	Fixed number of clusters	Very high	Moderate
Statistical models	F1 around 90%	Greater than 1,000	Fixed number of clusters	Very high	Moderate
Unsupervised algorithms	F1 around 70%	**Less than 3**	Fixed number of clusters	**No**	**Excellent**

The bold values highlight the best results for each column (metric).

algorithms, such as those applied in theproposed architecture. Table V summarizes the performance vs. the number of parameters, assumptions, data dependence (that tells the necessity of a training database), and, finally,the overall flexibility of different state-of-the-art models applied to story segmentation problems. We have analyzed quantitatively the properties of the architecture, against the most important approaches in the state of the art. Regarding data dependence, for our architecture we need a fewpieces of data to be able to tune the hyper-parameters of the algorithms. While for some supervised algorithms, most of them need only the number of clusters, so there is no dependency beyond the assumption. Note that we have not included [1] and [9] since they are used in the proposed architecture (GPA and SLM respectively). Transformer-based Deep Learning (BERT) models have been also considered within the proposed architecture (see Fig. 7(d)), however, they are NLP tokenizing models and only provide vector representations of the words, which are used by the architecture in a flexible way.

V. CONCLUSION AND FUTURE WORK

In this article, we have introduced an architecture that exploits spatial and temporal distance, and semantic correlation for optimal SSBN problems. Different algorithmshave been proposed to complete the flexible architecture and some recommendations have been made to implement each type of algorithm in the different architecture modules. Specifically, we have proposed a first algorithmthat is focused on merging consecutive sentences and detecting text borderlines between different pieces of news. We have also proposed a second algorithm focused on merging strongly correlated blocks, even if they are temporally and spatially distant. This architecture allows an easy plug-unplug of methods and models, which stronglydepends on the application and the language the model is oriented to. Additionally, a tree-based storage structure has been proposed to define a piece of news in a news broadcast, which stores information that defines the piece of news stored. We have evaluated the performance of the proposed architecture in the segmentation of real news pieces from Spanish Television, obtaining excellent results. This work opens new lines of research, such as developing new algorithms to be included in the proposed architecture or building up a new approach for news clustering between different newscasts, that allows performing studies about news persistence orthe temporal length of news pieces. The results obtained in the experiments carried out show that the proposed architecture performs well in detecting the boundaries of news segments, with high Precision, Recall, and F1 scores, as well as a low standard deviation and few outliers. However, there is still room for improvement in detecting the boundaries of news segments, particularlyfor the WD and Pk-score metrics. We have implemented the proposed architecture in an open-source Python Package in a GitHub repository [29], including both RTVE and Wikipedia cases to ease the replication of these experiments or take them to another level of complexity.

ACKNOWLEDGMENT

Special mention to Juan Antonio Carabaña, José Luis Ruiz-Mendoza, and Sonia Castillo for their support and assessment.

REFERENCES

[1] C.-H. Wu and C.-H. Hsieh, "Story segmentation and topic classification of broadcast news via a topic-based segmental model and a genetic algorithm," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 17, no. 8, pp. 1612–1623, Nov. 2009.

[2] H. Chen, L. Xie, C.-C. Leung, X. Lu, B. Ma, and H. Li,, "Modeling latent topics and temporal distance for story segmentation of broadcast news," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 25, no. 1, pp. 112–123, Jan. 2017.

[3] D. Cer et al., "Universal sentence encoder," 2018. [Online]. Available: https://arxiv.org/abs/1803.11175

[4] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013. [Online]. Available: https://arxiv.org/abs/1301.3781

[5] N. Stokes, J. Carthy, and A. F. Smeaton, "Segmenting broadcast news streams using lexical chains," Jul. 2002. [Online]. Available: https://doras.dcu.ie/324/

[6] N. Stokes, J. Carthy, and F. A. Smeaton, "Select: A lexical cohesion based news story segmentation system," *AI Commun.*, vol. 17, no. 1, pp. 3–12, 2004.

[7] X. Wang, L. Xie, M. Lu, B. Ma, E. S. Chng, and H. Li, "Broadcast news story segmentation using conditional random fields and multimodal features," *IEICE Trans. Inf. Syst.*, vol. 95, no. 5, pp. 1206–1215, 2012.

[8] H. Misra, F. Yvon, O. Cappé, and J. Jose, "Text segmentation: A topic modeling perspective," *Inf. Process. Manage.*, vol. 47, no. 4, pp. 528–544, 2011.

[9] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018. [Online]. Available: https://arxiv.org/abs/1810.04805

[10] P. L. Ranzato and M. Brambilla, "A text segmentation technique based on language models," Master's thesis, ING - Scuola di Ingegneria Industriale e dell'Informazione, Dec. 2019.

[11] T. Lattisi, D. Farina, and M. Ronchetti, "Semantic segmentation of text using deep learning," *Comput. Inform.*, vol. 41, no. 1, pp. 78–97, 2022.

[12] Z. Liu and Y. Wang, "TV news story segmentation using deep neural network," in *Proc. IEEE Int. Conf. Multimedia Expo Workshops*, 2018, pp. 1–4.

[13] J. Yu, L. Xie, X. Xiao, and E. S. Chng, "An end-to-end neural network approach to story segmentation," in *Proc. IEEE Asia-Pacific Signal Inf. Process. Assoc. Annu. Summit Conf.*, 2017, pp. 171–176.

[14] J. Yu, L. Xie, X. Xiao, and E. Chng, "A hybrid neural network hidden Markov model approach for automatic story segmentation," *J. Ambient Intell. Humanized Comput.*, vol. 8, pp. 925–936, 2017.

[15] J.-R. Yu and H. Shao, "Broadcast news story segmentation using sticky hierarchical dirichlet process," *Appl. Intell.*, vol. 52, no 11, pp. 12788–12800, 2022.

[16] A. Pérez, Emilio, T. Ariza, L. Gesuelli Pinto, and M. Mazuecos, "Parallel sentences dataset," 2022. [Online]. Available: https://huggingface.co/datasets/hackathon-pln-es/parallel-sentences/tree/main

[17] J. Cañete, G. Chaperon, R. Fuentes, J.-H. Ho, H. Kang, and J. Pérez, "Spanish pre-trained BERT model and evaluation data," in *Proc. PML4DC Int. Conf. Learn. Representations*, 2020, pp. 1–10.

[18] "Radio-Televisión Española," 2023. [Online]. Available: https://www.rtve.es

[19] M. Ester, H. P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proc. 2nd Int. Conf. Knowl. Discov. Data Mining*, 1996, pp. 226–231.

[20] W. Stuetzle, "Ward's method for hierarchical agglomerative clustering with application to gene expression data," *J. Comput. Biol.*, vol. 10, no. 2, pp. 155–173, 2003.

[21] J. Liu and S. Chaudhuri, "Efficient k-means-based clustering algorithms for large data sets," in *Proc. 16th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2010, pp. 611–620.

[22] K. Fukunaga and L. Hostetler, "The estimation of the gradient of a density function, with applications in pattern recognition," *IEEE Trans. Inf. Theory*, vol. 21, no. 1, pp. 32–40, Jan. 1975.

[23] A. Y. Ng, M. I. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *Proc. Adv. Neural Inf. Process. Syst.*, 2002, pp. 849–856.

[24] "Wikipedia [Website]," 2023. [Online]. Available: https://www.wikipedia.org

[25] P. May, "Machine translated multilingual STS benchmark dataset," 2021. [Online]. Available: https://github.com/PhilipMay/stsb-multi-mt

[26] L. Pevzner and M. A. Hearst, "A critique and improvement of an evaluation metric for text segmentation," *Comput. Linguistics*, vol. 28, no. 1, pp. 19–36, 2002.

[27] S. Lamprier, T. Amghar, B. Levrat, and F. Saubion, "On evaluation methodologies for text segmentation algorithms," in *Proc. IEEE 19th Int. Conf. Tools Artif. Intell.*, 2007, pp. 19–26.

[28] J. Farray, "Jacobo farray dataset," 2022. [Online]. Available: https://huggingface.co/jfarray/Model_dccuchile_bert-base-spanish-wwm-uncased_100_Epochs

[29] A. Palomo-Alonso, "NewSegmentation Python Package: Story Segmentation of Broadcast News," 2022. [Online]. Available: https://github.com/iTzAlver/NewSegmentation and https://pypi.org/project/newsegmentationRegenerateresponse

Alberto Palomo-Alonso was born in Alcalá de Henares, Spain, in 1998. He received the B.S. and M.S. degrees in telecommunication engineering in 2020 and 2022, respectively, from Universidad de Alcalá, Alcala de Henares, Spain, where he is currently working toward the Ph.D. degree in natural language processing with the Department of Signal Processing and Communications. He has worked on several artificial intelligence and optimization problems. He has also participated in projects related to blockchain and embedded systems oriented to telecommunications and other signal-processing projects. His main research interests include development of machine learning, hybrid, statistics, and RL algorithms, and their applications in natural language processing and other fields of technology and science.

David Casillas-Pérez received the B.S. degree in telecommunication engineering and the M.S. and Ph.D. degrees in electronic control systems from Universidad de Alcalá, Alcala de Henares, Spain, in 2013, 2014, and 2019, respectively. He is currently an Assistant Professor with the Department of Signal Processing and Communications, Universidad Rey Juan Carlos, Madrid, Spain. His research focuses on the development of machine learning algorithms with applications in different fields such as computer vision, mobile communication systems, and renewable energy systems.

Silvia Jiménez-Fernández was born in Madrid, Spain, in 1976. She received the B.S. and Ph.D. degrees in telecommunications engineering from Universidad Politécnica de Madrid, Madrid, Spain. She is currently an Associate Professor with the Department of Signal Processing and Communications, where she researches on the application of signal processing and machine learning techniques for mobile communication systems.

Jose A. Portilla-Figueras was born in Santander, Spain, in 1976. He received the B.S. degree and the Ph.D. degree in telecommunications engineering from Universidad de Cantabria, Santander, Spain, in 1999 and 2004, respectively. He is currently a Full Professor with the Department of Signal Processing and Communications, and the Head of the Polytechnic School of Universidad de Alcalá, Alcala de Henares, Spain. His research interests include mobile communications systems, 5G systems, and the development of machine learning algorithms with applications in telecommunication engineering problems.

Sancho Salcedo-Sanz was born in Madrid, Spain, in 1974. He received the B.S degree in physics from Universidad Complutense de Madrid, Madrid, Spain, in 1998, the Ph.D. degree in telecommunications engineering from the Universidad Carlos III de Madrid, Madrid, in 2002, and the second Ph.D. degree in physics from Universidad Complutense de Madrid, in 2019. He spent one year with the School of Computer Science, University of Birmingham, Birmingham, U.K, as a Postdoctoral Research Fellow. He is currently a Full Professor with the Department of Signal Processing and Communications, Universidad de Alcalá, Alcala de Henares, Spain. He has co-authored more than 225 international journal papers in the field of machine learning and soft-computing. His research interests include soft-computing techniques, hybrid algorithms and neural networks in different applications of science and technology.