CentraleSupélec

Cursus Ingénieur 3A

Dominante Mathématiques et Data Science

Environmental Communication Index

Auteurs:
Antoine Dargier
Martin Lanchon
Martin Ponchon
Alexandre Pradeilles

Superviseurs:
Pietro Turati
Arnaud Benoits
Jean Sauvignon

April 19, 2023

Abstract

This project aims to quantify the share of environmental topics in the French media. To do this, we used an Open API from Radio France and scraping to collect radio broadcasts, which we transcribed using the Wav2Vec2.0 model, on an Azure virtual machine. In parallel, we scraped millions of press articles, and developed a text classification tool based on CamemBERT, achieving 92% accuracy. We developed a streamlit visualization platform to highlight these results and allow everyone to follow the evolution of environmental topics.

Contents

1 Introduction

We are a group of four students in our final year at CentraleSupélec, in the Data Science section of the Mathematics and Data Science department. As part of this year, we have a long project to carry out with a company. We decided to choose the project proposed by Eleven Strategy, a strategy and digital consulting firm. It is about quantifying the share of environmental topics in the French media. Behind this subject, state-of-the-art NLP methods will be applied, and their interest will be shown to the firm. We chose this topic first of all for its impact: all four of us are very concerned with environmental issues, and we liked to participate in highlighting problems or good practices in the media. Moreover, this topic was very interesting to use very recent NLP methods, which we had never used in other settings. Finally, this project is also interesting and exciting because we are working on the whole chain, from data ingestion to visualisation. We were therefore able to use and discover many tools, which was very instructive for us.

2 Project Organization and Management

2.1 The commissioner

The project was proposed by Eleven Strategy, a French consulting firm.

eleven
strategy consultants
London • Paris • Madrid

Eleven is the first strategy consulting firm specifically founded to assist companies in their adaptation to digital, AI (artificial intelligence) and digital transformation. To achieve this, eleven relies on a unique combination of strategic analysis, an entrepreneurial approach and strong proximity to the digital and data ecosystem.

The project was led by three consultants of the firm: one partner, Pietro Turati, and two consultants: Arnaud Benoits and Jean Sauvignon. The goal of this project was to use an important subject, media coverage of climate subject, to work on state-of-the-art NLP algorithms, which have a lot of value for the firm. Indeed, the project allows us and eleven Strategy to work on NLP classification models, speech-to-text. Moreover, it is a end-to-end project, where we created a showcase dashboard for the company on a very important subject which is the environment. It also shows the company's commitments and values.

2.2 Project organization and schedule

For the launch of the project, we met Arnaud, Jean and Pietro in real, to get to know each other, decide together exactly what we want to do, and how we will work. That allows us to meet eleven's employees and discover their Paris office. We decided very quickly to have meetings every two weeks, on Wednesday, to speak about our achievements, work, problems and so on. We schedule these meetings online via Teams because it was easier for everybody, not to have to make the journeys between Paris and Gif-sur-Yvette. As the project days were on Wednesdays, this was the right rhythm to follow the project regularly and to have interesting progress.

After an overview of the work required, we launched two streams of work in parallel, to deal with the two main sources of data that we chose to focus on:

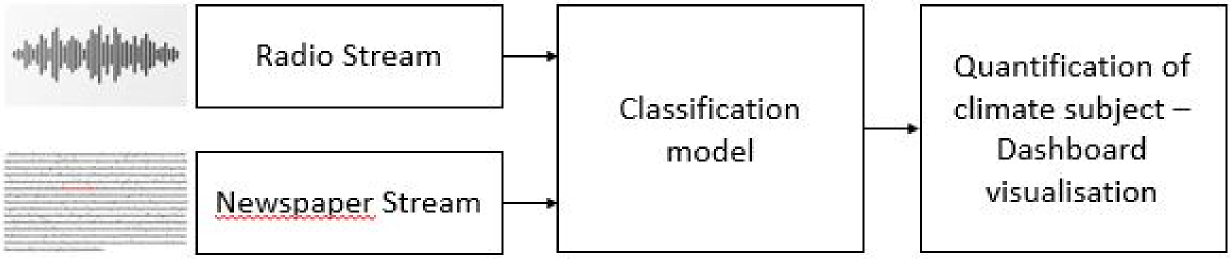
- Text data from newspapers
- Audio data from radio

For both streams, an extensive literature review was conducted until December, to fully understand the subject, the different possible methods and the state of the art. Then, the radio stream worked until March to test the models, acquire the data, transcribe and set up the pipeline on the virtual machine. During this time, the newspaper stream worked on the NLP classification models, their implementation and fine-tuning, and on the scraping of various newspapers (20 minutes, Libération, etc.). Finally, from mid-March, we started to develop the dashboard, and our report and presentation.

2.3 Decision-making process and distribution of work

About the , we were able to define the outlines of the project together, see what we wanted to do, the subjects we wanted to deal with, ... We had a lot of freedom, which was very nice to do what we wanted. Thus, Antoine and Martin P. worked on the radio stream while Alexandre and Martin L. worked on the log stream and the classification algorithm. Then, we all worked together on the implementation of the platform. To summarise, here is the organisation of our project:

Figure 1: Project organization



3 Automatic Speech Recognition

In this part, we will study how we deal with radio emissions. Our goal was to transcript the emissions to get the text, and then be able to use the NLP classification method to find the subject of emissions. In the introduction, we will see the interests in this field of research, its history briefly, the current state of research, some examples and limits to overcome. Then we will detail our methodology, ie the different models we tried and their performances. After the choice of the model, we will detail the data used for the experimentation, and finally all our pipeline of transcriptions.

3.1 Introduction

Automatic Speech Recognition (ASR), also known as speech-to-text, consists in converting spoken language as an audio signal into written text. While easy for humans, this task has historically been hard for machines. ASR is a very dynamic field of research because of its utility in our daily life. Speech-to-text enables better human-to-machine interaction and has numerous applications.

Research into speech recognition began in the mid-twentieth century. The first system to enter this field appeared in 1952 and was able to recognise the 10 digits. After that, research improved greatly with IBM and the work of Jelinek in the 1970s. In 1972 the first word recognition system was marketed by the company Threshold Technologies. This science experienced a new boom in the 2000s when Google added a voice-activated search tool to its search engine in 2008 and Apple added Siri to its phones from 2011. In 2017 Microsoft announced that it had achieved human voice recognition performance.

The topic is significant because we are then able to obtain a lot of information about the speech from the text: the content, the topic, the interlocutors, the tone, ... Speech recognition can be related to many areas of science: automatic language processing, linguistics, information theory, signal processing, neural networks, artificial intelligence, etc. Among the various fields of research in this area, speech recognition, as well as speech synthesis, speaker identification or speaker verification, are among the most important speech processing techniques. The GAFAM have well understood the potential of the technology, and work on algorithms for years, which are already applied to their products, such as Siri for Apple, Alexa for Amazon, the Google Home, Cortona for Microsoft ... The fields of application of these tools are enormous: help for people, assistance and presence for elderly or disabled people, a source of information, and a way to store information, ... This could have a direct impact on professions such as court clerks or translators. It is therefore a real technological innovation with a significant impact. However, the current solutions are not yet satisfying, because there are still important errors in real environments. Algorithms still have problems when the speech is in a noisy environment and when several persons speak simultaneously. It's a challenge too, to be sure to construct meaningful sentences when creating the text. In addition, researchers such as W. Minker and S. Bennacef, in their publication Speech and Human-Computer Dialogue, have confirmed the complexity of these models by the great difference between formal language understood by machines, and natural language. In formal language, there are strict and unambiguous rules of grammar, spelling and syntax. In natural language, evolutions can occur in the meaning of words, their use, and the message can be very different depending on the context or intonation. It is very difficult for a machine to perceive this.

3.2 Speech-to-text models

During our review of state-of-the-art Speech to text models, we discover different methods developed to get the text from audio. First, we analyse traditional methods, and then we try to implement and use three models: a Convolutional Neural Network model, a Connectionist Temporal Classification model and finally Wav2Vec2.0, a pre-trained model using transformers. We will detail here how these models work, and then we will see their performances on a French dataset.

3.2.1 The traditional ASR pipeline

We have begun our study by trying to understand the traditional ASR pipeline. The steps of implementation are described by Preethi Jyothi for Microsoft in a video in 2017 [1].

The traditional pipeline is composed of several independent components, essentially breaking down the task of speech recognition into building an acoustic model, a pronunciation model and a language model. Another characteristic is the decomposition of words into phonemes (distinct units of sounds). This allows the construction of words based on their pronunciation.

In the first step, the audio signal is converted into a data frame with acoustic features with a Fourier Transform. Different models are then applied successively to transform the data in text:

Acoustic Model: This model is trained to relate audio data (frequencies, intensity with time) to phonemes. This model estimates the probability $P(X|H)$ of the observed sequence of acoustic vectors X given a possible pronunciation H of a given word sequence.

Pronunciation Model: This model is able to create words from an association of phonemes. For example, given the phonemes [g], [uh], and [d], the model is able to understand that it's the word good. This model gives for each sequence of words W, the possible pronunciations H with their probabilities $P(H|W)$.

Language Model: It creates meaningful sentences from an association of words. This model gives the probability $P(W)$ of each word sequence W in the target language.

The problem is to find the sequence of words that has the highest probability. Formally, the problem amounts to finding the sequence of words W that maximises the following function:

$$P(W) \sum_{H} P(H|W)P(X|H)$$

More recently, with the advent of deep learning, to try to improve the ASR pipeline, researchers have started by taking one of these core components and replacing them with a deep learning algorithm. For instance, [2] has proposed to replace the classical Gaussian Mixture acoustic model with a Deep Belief Network with some pre-training strategies, leading to a more than 10 percent increase in accuracy.

While flexible, this pipeline faces major issues: Since each component is trained independently from the others with different objective functions, errors in one block may not behave well with other errors, which makes the task of improving the accuracy particularly difficult. Moreover, building each of these components from scratch requires significant work. For these reasons, current research now focuses on building end-to-end models. We have therefore chosen to work on these types of models, for which there have been major developments in the past few years.

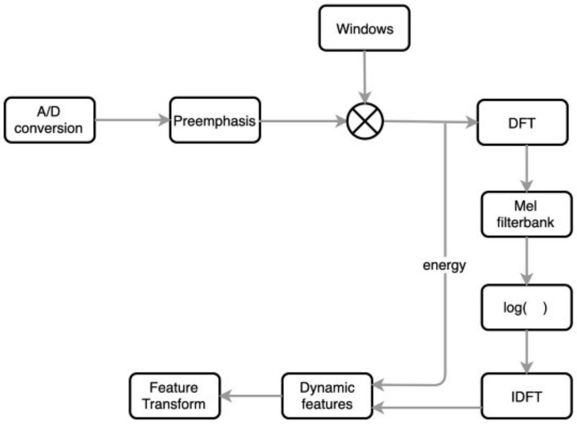
3.2.2 Our first CNN model

Our first approach is to use a Convolutional Neural Network (CNN).

The raw audio signal is not suitable to use as input for our model because it contains a lot of unwanted noise. Instead, we can improve the performance of the model by extracting important features from the audio signal and using those as input. One commonly used method for extracting features from audio is called Mel-Frequency Cepstral Coefficients (MFCCs).

MFCC is a feature extraction technique commonly used in speech and audio processing. It converts the audio signal from the time domain to the frequency domain and then applies a non-linear transformation to represent the audio in a way that is more similar to how humans perceive sound. MFCC is a representation of the short-term power spectrum of a sound. The road map of the MFCC computation is the following:

Figure 2: Road map of the MFCCs computation



MFCC values, which are extracted from audio signals, can be affected by noise. To make them less sensitive to noise, it is common to normalize their values in speech recognition systems. Some researchers have suggested modifications to the standard MFCC algorithm in order to make it more robust to noise.

The resulting coefficients, which are a set of numbers, can be used as input to a machine learning model for tasks such as speech recognition, speaker identification, and music genre classification.

The model consists of a sequence of layers of neurons, each with a goal in the network:

- Convolutional layers: responsible for detecting patterns and features in the input data
- Pooling layers: used to reduce the dimension of the data and control overfitting
- Flatten layer: used to convert the multi-dimensional array output from the previous layers into a one-dimensional array, so that it can be used as input for the dense layers
- Dense layer: responsible for making the final predictions based on the output of the previous layers
- Dropout layer: Dropout is a regularization technique used to prevent overfitting in deep learning models. It works by randomly dropping out (setting to zero) a certain percentage of neurons during training, so that the model cannot rely on any one

feature too much. This helps to reduce the chance of the model memorizing the training data, instead of generalizing to new data

Classical CNN models are trained using backpropagation and gradient descent.

We have trained our model on Google Colab, which allows access to Kaggle datasets without downloading them locally. We were able to use their computing power to train our model for 6 hours on 2,000 audios. This represents 5h30 of training audios.

3.2.3 Connectionist Temporal Classification

Connectionist Temporal Classification [3], more commonly referred to as CTC, is an end-to-end system introduced to address the alignment problem, which is one of the main issues in speech recognition. This difficulty arises because audio inputs of variable lengths may need to be mapped to the same transcription depending on the speed at which a person is speaking. CTC is nowadays often used in combination with other techniques in speech recognition tasks.

This model is often implemented as a grapheme-based model, i.e. outputting characters directly, while traditional pipelines were based on a phoneme decomposition of words. In addition to the letters of the alphabet (including accents depending on the language), the model uses a blank character (denoted by an underscore _ below) and an unknown character.

The architecture relies on a Recurrent Neural Network, trained on labelled data (i.e. transcribed audio):

1. Raw audio is pre-processed to obtain a simple spectrogram (which corresponds to calculating Fast Fourier Transforms on small 20 ms windows of waveform) to get frequency content.

2. Frames go through the RNN. Output neurons of the RNN are softmax neurons encoding a distribution $P(c_i|x)$ over symbols. Under independence assumption,

$$P(c|x) = \prod_{i=1}^{N} P(c_i|x) \quad \text{with } c_i \in \{A, B, ..., \text{blank}\}$$

   Note that c is a sequence of symbols that is of the same length as the audio.

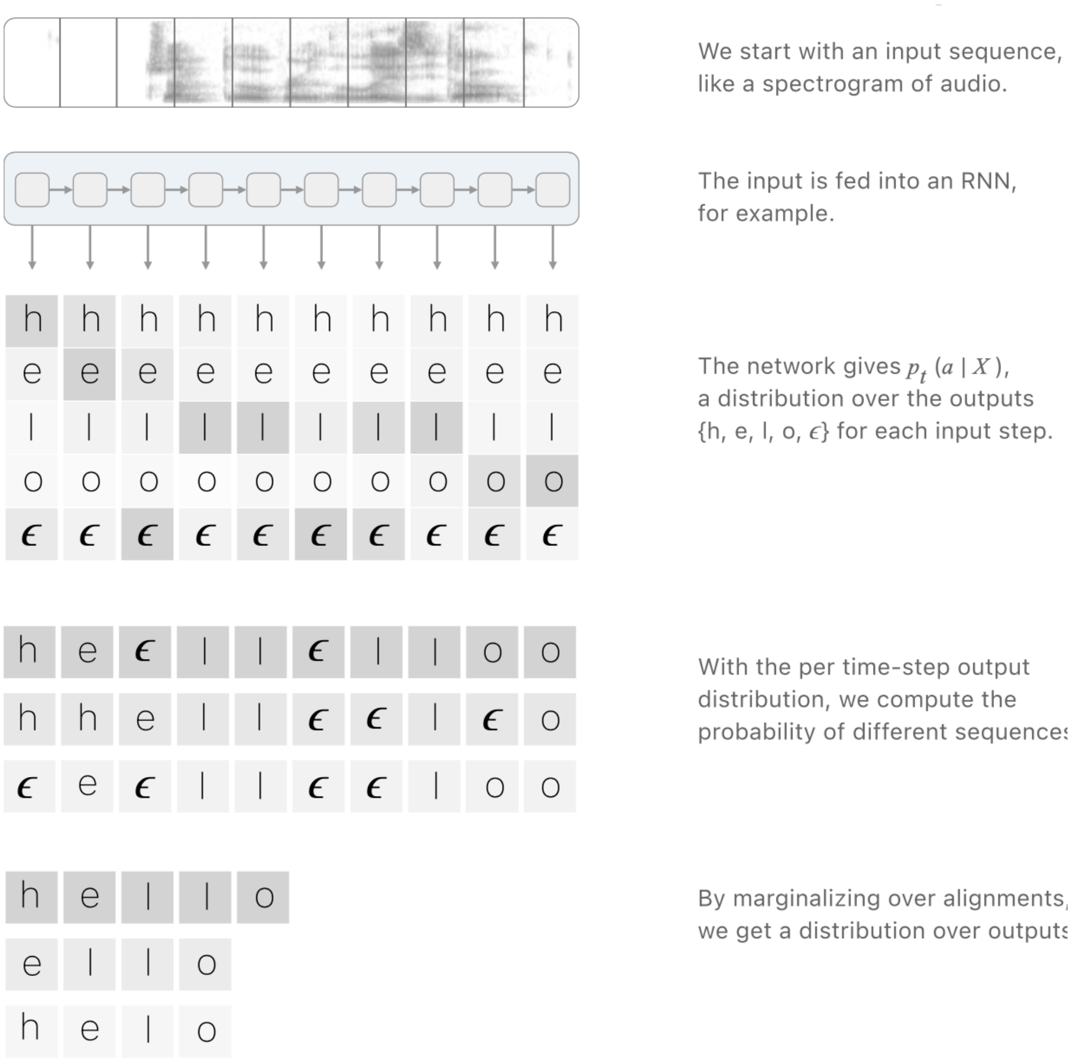
3. A mapping β takes as input the encoding c and returns the transcription y. β essentially removes duplicates and blank characters: For example, $\beta(HHH\_EE\_LL\_LO) = HELLO$. Since several c can lead to the same y, we marginalize to get a distribution over transcriptions

$$P(y|x) = \sum_{c:\beta(c)=y} P(c|x)$$

Network parameters are updated with backpropagation to maximize the likelihood of choosing the correct label

$$L(\theta) = P(y^{*(i)}|x^{(i)}) = CTC(c^{(i)}, y^{(i)})$$

Figure 3: CTC Overview


We start with an input sequence, like a spectrogram of audio.
The input is fed into an RNN, for example.
h e l o ϵ
The network gives $p_t(a \mid X)$, a distribution over the outputs {h, e, l, o, ϵ} for each input step.
h e ϵ l l ϵ l l o o
h h e l l ϵ ϵ l ϵ o
ϵ e ϵ l l ϵ ϵ l o o
With the per time-step output distribution, we compute the probability of different sequences
h e l l o
e l l o
h e l o
By marginalizing over alignments, we get a distribution over outputs

While CTC as an end-to-end system described above is no longer the state-of-the-art, CTC loss is the objective function used for fine-tuning on labelled data proposed by [4] to perform speech recognition.

3.2.4 Wav2Vec2.0

Wav2Vec 2.0 is a state-of-the-art end-to-end audio pre-processing technique used for speech and audio recognition tasks. It was published in 2019 and it is an extension of the original Wav2Vec model, which was developed by Meta. The scientists behind the model are Alexei Baevski, Henry Zhou, Abdelrahman Mohamed and Michael Auli.

This model has many advantages over older models. First of all, the model is based on a self-supervised algorithm. It therefore does not require supervised data for training. This is extremely beneficial in the case of ASR as it allows the learning of many languages, which was not previously possible. Indeed, in previous models, a lot of audios were needed with their transcription, which is not available in large quantities for all languages except English. For Wav2Vec2.0, the model needed 50,000 hours of audio for training, which would have been difficult with the transcripts. In the second phase, very little labelled data with the transcripts are needed to get greatly improved results: 10 minutes of labelled data was enough.

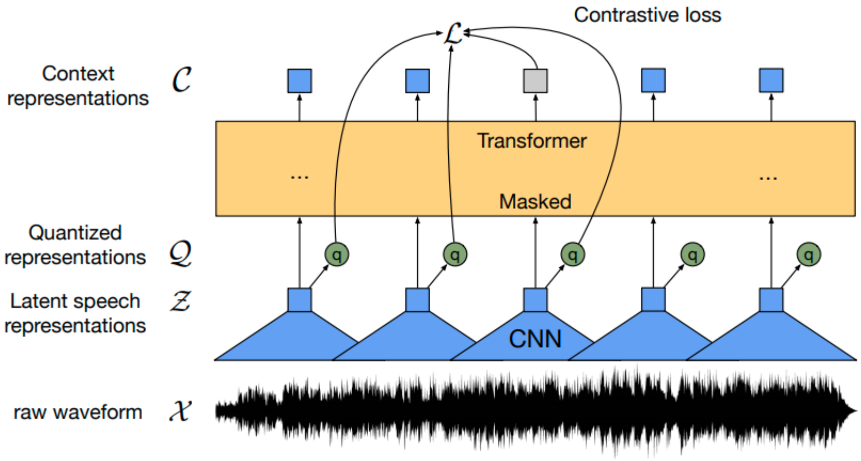
In addition, there are many packages available today that make using Wav2Vec2.0 quick and easy. We were able to use HuggingSound from HuggingFace, and the implementation was very fast.

Wav2Vec 2.0 is a self-supervised model, meaning it does not require labelled data for training. Instead, it is trained on a large dataset of unlabelled audio files and learns to extract useful features from the audio by predicting the next audio frame in a sequence. This is done using a Contrastive Predictive Coding (CPC) technique. It is a technique used to learn a compact and semantically meaningful representation of the data. This is done by training a model to predict the next frame of an input sequence. The CPC technique consists of two main components: an encoder and a predictor. The encoder processes the input sequence and generates a fixed-length representation of the sequence, which is called the ”context vector”. The predictor then tries to predict the next frame in the sequence using the context vector as input.

There are three main steps in the construction of the algorithm:

- CNN with 7 layers to learn the latent speech representations Z
- Quantization: try to match the Z with codebooks, sort of phonemes
- Transformers to learn contextual representation

Figure 4: wav2vec2.0 architecture



The feature encoder is based on a temporal convolution followed by a GELU activation function.

For the training, noise is added to the audio so that the model is able to distinguish the speech and the noise. It really improves the performances of the algorithms in all contexts. The loss function to minimize is then the composition of 3 functions:

$$Loss = L_{contrastive} + L_{diversity} + L_{L2} \tag{1}$$

where

- $L_{contrastive}$ measures the similarity between the predicted next frame and the actual next frame in the sequence. It is calculated as the dot product of the predicted frame and the actual frame: $L_{contrastive} = -\log(\exp s(x, x')/(\sum \exp s(x, x')))$, with s the similarity function, often chosen to be a dot product
- $L_{diversity}$ is used to encourage the model to generate a wide variety of different outputs, rather than simply memorizing the training data. It measures the dissimilarity or diversity between the generated samples and the real samples
- L_{L2} is used to prevent overfitting in machine learning models by adding a term to the loss function that penalizes certain model parameters. Also known as Ridge regularization, it adds a term to the loss function that is proportional to the square of the model weights. It tends to produce models with small but non-zero weights.

Finally, for the encoder, two language models were trained: a 4-gram model coming from NLP methods and a transformer.

The model was trained on CommonVoice, which is the biggest dataset with audios and transcriptions in French. Then, two pre-trained were created: a BASE (with a dimension of 768) and LARGE (dimension of 1024).

As said in the advantages part, the model was trained on 53,000 hours of unlabelled samples and only 10 minutes of labelled data were needed. For that training, the teams from Meta used the LibriVox and LibriSpeech datasets.

3.3 Evaluation & results

To test our algorithms and Wav2Vec2.0, we searched for a French dataset with audio and transcriptions. We chose the Kaggle dataset by Kyubyong Park and Tommy Mulc named ”French Single Speaker Speech Dataset” [5]. This dataset has 8,600 audios of about ten seconds each, which represents 24 hours of recording. The recordings are of single people reading sentences from the novels Les Misérables by Victor Hugo and Arsène Lupin contre Herlock Sholmes by Maurice Leblanc. From a speech-to-text point of view, we are therefore in the easiest case, working with read text without noise or several people speaking at the same time.

The most common metric in speech-to-text is the Word Error Rate (WER), which measures the difference between the predicted words and the actual transcription. It takes into account substitutions, insertions and deletions. Thus, we can define the WER by the equation:

$$WER = \frac{S + I + D}{N} \tag{2}$$

where

- S = number of substitutions (word replaced by another in the prediction)
- I = number of insertions (word added in the prediction)
- D = number of deletions (word removed in the prediction)
- N = number of words in the reference

Thus, the WER value varies between 0 and can exceed 100%, because the number of insertions is not limited.

When testing methods, it is always important for this kind of study to try on a variety of data. Thus, researchers often test on clean data and on noisy data.

Here are some average results for the WER:

- read texts (voice dictation, single-talker system): $WER \approx 5\%$
- radio and TV news: $WER \approx 10\%$
- informal telephone conversations: $WER \approx 40\%$

It exists different models use nowadays, and to compare their performance, it's important to test them on different datasets. Indeed, the environment, the noise, the number of speakers, etc. can largely influence the result. In their analysis published on GitHub on February 2022 [4], Picovoice (one actor of ASR) try to evaluate and compare the performances of the different main algorithms. For that, they average the results on 4 datasets: LibriSpeech clean, LibriSpeech other, TEDLIUM and CommonVoice. Here are the results they obtain for such algorithms:

- Azure: $WER = 7.93\%$
- Amazon: $WER = 8.74\%$
- Google Enhanced: $WER = 11.32\%$
- Google: $WER = 20.46\%$
- IBM: $WER = 22.04\%$
- Mozilla DeepSpeech: $WER = 22.86\%$

Concerning the CNN model, it was very difficult to train it with our processing capacity. In fact, the model almost always predicts the same words and sentences, and was not too large or not enough trained to have better results. So, the WER was very bad, about 80%. For this type of model with more training, we found in the literature WER above 20% [6].

Starting from [2] we tried to implement our own CTC model but were unable to do so due to insufficient processing power. According to [2], training for more than 50 epochs would be necessary to obtain acceptable results, and each epoch would take 5 to 6 min to train with a GPU (which we don't have). Under these assumptions, they achieved a 16% WER.

Concerning Wav2Vec2.0, the last results published were very impressive, and improve a lot the WER comparing to the other models. In the first paper, the authors announced WER of 5.7% on clean audio and 10.1% on noisy audio. In the last paper, the results were even better: they reached 1.8% WER on the test-clean subset and 3.3% WER on the test-other.

We wanted to compare that results and compute our own results. We used Wa2vec2.0 pre-trained models and tried to predict the transcription of 580 audios of our dataset. That represents about 1h30 of transcriptions. With that, we obtain a WER of 21.7%. The results are not as good as in the paper, but in large part because of the punctuation. The Wav2Vec2.0 algorithm predicts sometimes punctuation, such as commas and apostrophes, but not always. We couldn't thus decide whether to take the punctuation into account or not, and it increases the difference between the predictions and real transcriptions.

Here is a little review of all the performances of the models. We were not able to test all methods on our dataset because they are not free for lot of them. Moreover, there are sometimes black boxes, which can be less interesting for our analysis.

Table 1: Performances of Speech-to-text models

Model	WER in litterature	WER on our dataset
Azure	7.93%	-
Amazon	8.74%	-
Google Enhanced	11.32%	-
Google	20.46%	-
IBM	22.04%	-
Mozilla DeepSpeech	22.86%	-
Our CNN	20%	80%
CTC	16%	-
Wav2Vec2.0	$1.8 - 3.3\%$	21.7%

Finally, according to these results, we chose to use Wav2Vec2.0, which seems to have the best results with our French dataset.

3.4 Use of radio audio

Now that we have our model, we need to get the French radio data. To do this, we were able to access the OpenAPI of Radio France, which is freely available on the condition that you request a key. On this API, it is possible to find all the broadcasts of the French radio stations (FranceInfo, FranceInter, FranceBleu, ...). We can follow live broadcasts, find past broadcasts, and have information such as the title of the broadcast, the date of broadcast, and the production, ... So we used this tool to get information about the URL of the shows. From the URL, we could download an mp3 file that we could then transcribe with our template. However, we could not go very far back with this method, because the URL of the mp3 files are quickly deleted. So we scrapped directly the podcast sites of the radios to get this URL.

3.5 Transcription pipeline

Now we have our model and the data, all that remains is the transcription. We realised that the transcription step was very time-consuming, as our Wav2Vec2.0 model is real-time: it takes about 10 seconds to transcribe a 10-second audio (real-time factor = 1). With eleven Strategy, we, therefore, chose to use a virtual machine on the Azure environment to carry out this operation, using multi-processing to make full use of the machine's power.

In addition, we chose to focus on a single news programme: the 7-9h on FranceInter. This programme seemed relevant to us, with topical subjects and political or artistic guests. We transcribed one programme per week, in order to be able to go back far enough and see if any trends are emerging. In the end, we were able to transcribe 218 programmes of 2 hours or 2.5 hours, between 23 December 2016 and 15 March 2023.

Figure 5: Emission chosen for the transcriptions

4 Text Topic Classification

This part is dedicated to the Text-topic-classification technology brick. Text-topic classification is a NLP task that involves categorizing a piece of text into predefined categories or topics based on its content. After a short introduction, we will study the data and metrics used for this model. Then we will review the different models tried in order to present our final model.

4.1 Introduction

A fundamental technological brick in the Environmental Communication Index is the classification of the items we obtain. Text-Topic-Classification involves assigning predefined topics to a given text document.
Given a set of documents, $D = d_1, d_2, ..., d_n$, and a set of predefined topics, $T = t_1, t_2, ..., t_m$, the problem is to find the topic t_j that best describes each document d_i.

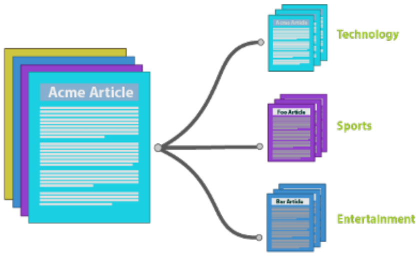


Figure 6: Text topic classification for newspaper

4.1.1 Main related works

Over the years, various approaches have been proposed for text topic classification, ranging from rule-based and statistical methods to more recent deep learning techniques.

Early approaches to text topic classification relied on handcrafted rules and heuristics, such as keyword-based methods, which used a list of pre-defined keywords to assign labels to a document. Other rule-based methods used features such as n-grams, term frequency, and inverse document frequency (TF-IDF) to classify text documents. Statistical methods, on the other hand, made use of machine learning algorithms to automatically learn features from text data and use them to classify documents. Popular statistical methods for text classification include the Naive Bayes algorithm and the Support Vector Machines (SVM) algorithm.

One of the limiting factors of classical models is their reliance on explicit feature extraction procedures. Good feature engineering requires extensive domain knowledge, which in turn, makes it difficult to generalize approaches to new tasks. Furthermore, manual feature crafting does not utilize the full potential of large amounts of training data because of how features are predefined rather than discovered.

For these reasons, Deep Learning has progressively imposed itself in NLP. The first Text-topic-classification models based on Recurrent Neural Network, Convolutional Neural Networks, Graph Neural Networks have given way to Transformer-like architectures. The most recent language models such as BERT and GPT achieve excellent performance and all Datasets. BERT and GPT were just the starting point for the development of many variations and improvements, of which we cite the most influential. Robustly Optimized BERT Pretraining Approach (RoBERTa) [7] was introduced by Liu et al. as a successor of BERT. The GPT model has also had successors, the most popular being the GPT-3 and recently the GPT-4 model.

4.2 Dataset

The first step was to create our dataset. We needed to have the whole archives from several newspapers in order to be able to follow the evolution of the number of articles about climate change. We have then selected three freely available newspaper: 20 Minutes, Liberation and

Le Point.

We have used the Python framework Scrappy with has allow us to extract all the articles from the websites of these three companies. For each article, we have extracted the title and the body to be able to make predictions, but also many metadata such as the image URL, the date or the journalist to give as much context as possible, but we also have extracted the category the article belong to in order to have a label for each article as it would have been really long to manually labeled this quantity of articles.

We also annotated a few hundred items later in the project. Indeed, in order to explore Multi-class models (see section 4.5), we had to create our own annotations.

Table 2: Number of articles scraped from newspapers

Newspaper	Number of articles
20 Minutes	977,935
Libération	403,916
Le Point	907,115

4.3 Text-topic-classification models

4.3.1 Unsupervised learning with Doc2Vec

This model acts as a baseline model. We created embeddings thanks to the Doc2Vec architecture in order to get a vector representation of each article. Doc2vec is an extension of the word2vec algorithm which represents each word by a vector depending on what other words are often found around it, which make words used in a similar context really closed in the embedding space. Then we chose several cluster equal to the number of categories we are interested in and perform a Gaussian Mixture Model to get our classes. As expected, the unsupervised model achieves low accuracy ($\sim 50\%$) because nothing predestines the clusters to match the categories.

4.3.2 Lbl2vec architecture

We tried to improve this model with the Lbl2vec architecture [8]. Categories are manually defined which allows the creation of consistent cluster centroids by giving some keywords for each class. Then we train a Doc2Vec on the entire dataset. This allows us to extract the vector representation of each keyword, then by pooling them we obtain a category representation. Then we just must assign each article to its closest category according to their representation vectors.

This model perfomed better than the previous one because the categories are now defined (Accuracy of 75%). However, it is still very sensitive to category definitions.

4.3.3 Pretrained CamemBERT

The next step was to use a CamemBERT with a classification head finetuned on our dataset. CamembBERT is one of the most successful pre-trained language models for French language processing. This model is a French variant of the BERT model that has been pre-trained on large French text corpora [9].

CamemBERT-base (the base version of the model) is composed of:

- an embedding layer to represent each word as a vector
- 12 hidden layers composed mainly of two types of transformations: self-attention transformations and dense transformations

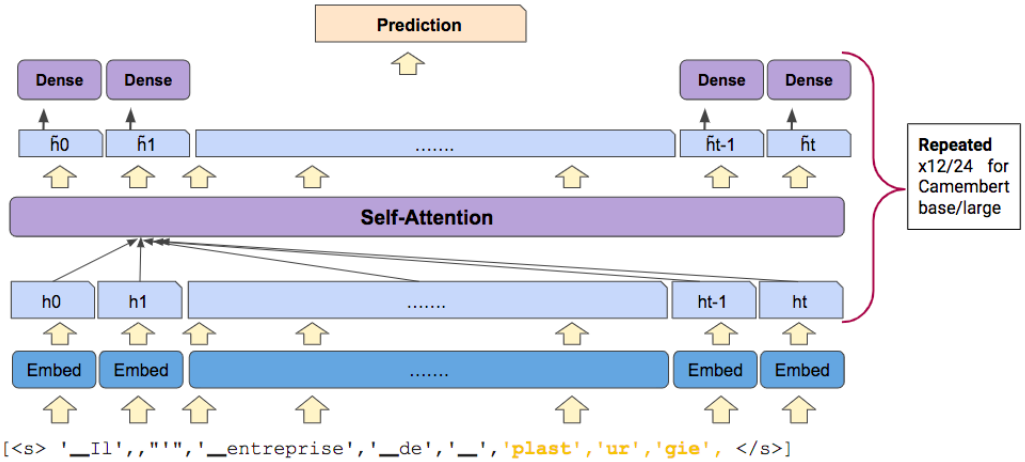

Prediction
Dense
Dense
Dense
Dense
h̃0
h̃1
....
h̃t-1
h̃t
Repeated x12/24 for Camembert base/large
Self-Attention
h0
h1
....
ht-1
ht
Embed
Embed
....
Embed
Embed
[<s> '__Il',,"'",'__entreprise','__de','__','plast','ur','gie', </s>]

Figure 7: CamemBERT Architecture

BERTs models are Transformers-based models. Transformers have been introduced in 2017 in the paper "Attention is all you need" [10]. But how does attention work? Let's say we want a contextual embedding for the word "bank" in the sentence "They cashed a check at the bank yesterday". The contextual vector of bank can be computed as a weighted sum of all input token vectors, e.g.:

$$h_{bank} = 0.01e_{They} + 0.25e_{cashed} + 0.007e_{a} + 0.31e_{check} + 0.06e_{at} + 0.01e_{the} + 0.35e_{bank} + 0.0001e_{yesterday}$$

Each weight represents the relevance of the considered token to the representation of bank. The weights are normalized to the interval $[0, 1]$ and are token-specific (e.g. weights for h_{bank} are not the same as those for h_{cashed}).The weights are not (necessarily) symmetric:

relevance of check for bank is not necessarily the same as the relevance of bank for check. More generally

$$h_i = \sum_{1 \leq j \leq t} \alpha_{i,j} e_j$$

(Non-normalized) attention weights $\alpha'_{i,j}$ are a function of x_i and x_j:

$$\alpha'_{i,j} = x_i^T x_j \in \mathbb{R}$$

where the (x_i) are the vectors of the word in the sentence. The more "similar" the two vectors are, the larger will be the result of the dot product.
They are then normalized with a softmax:

$$\alpha_{i,j} = \frac{\exp \alpha'_{i,j}}{\sum_{1 \leq j \leq t} \alpha'_{i,j}}$$

and finally:

$$h_i = \sum_{1 \leq j \leq t} \alpha_{i,j} x_j$$

4.3.4 Zero-shot learning

Finally, based on a pretrained CamemBERT on HuggingFace, we tried zero-shot classification [11] by comparing the embedding of each article and the embedding of sentences representing each category. The sentence used is "The article is talking about category_name.".
Then we are able to compute the cosine similarity between the embeddings of the article and each class. And we can transform these values into probabilities by normalizing them. Thanks to these probabilities, we are able to perform normal classification or multi-labels classification as we will see now.

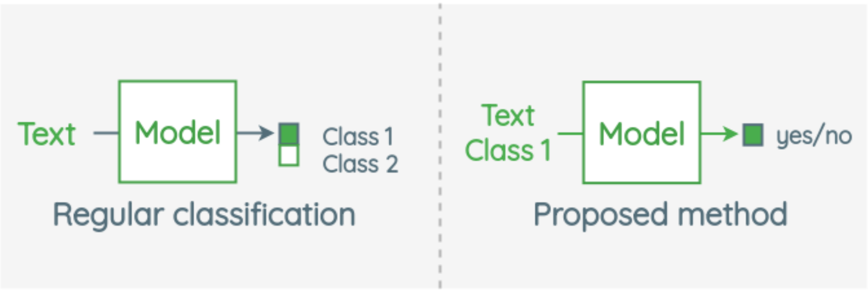
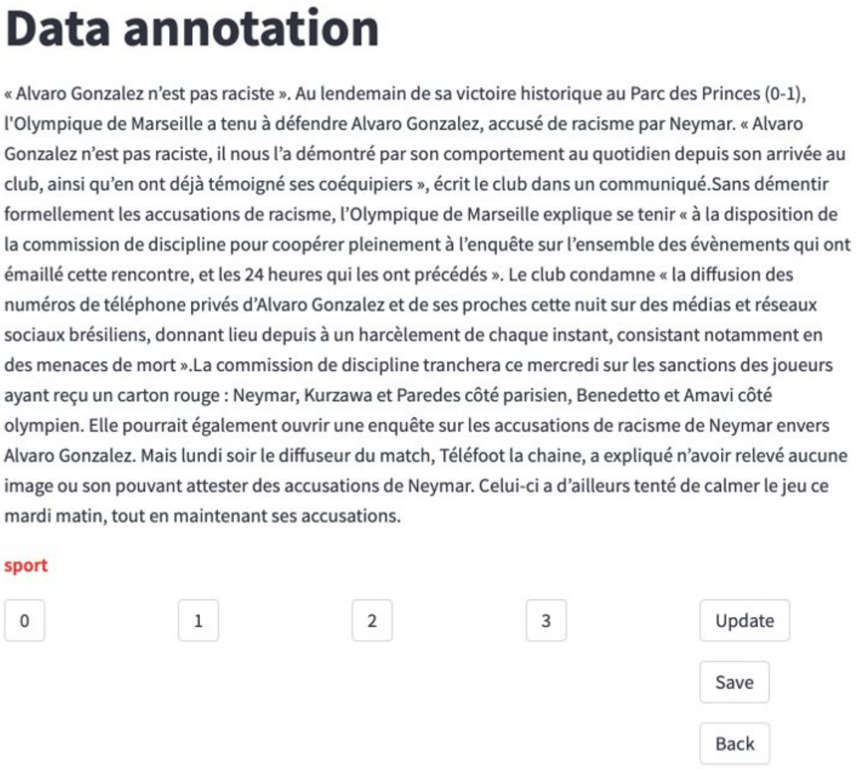


Figure 8: Zero-shot vs regular classification

4.4 Multi-categories classification

While reading some articles scrapped and doing our experiments, we have noticed that most of the article even if the newspapers classify them in only one category, could in fact belong to several classes.
To deal with this issue and improve the quality of our predictions, we have decided to perform multi-label classification. However, the training part of our model have not change

as the newspaper categories are relevant and we have a lot of annotated data, it was not possible to annotate enough multi-labeled articles to perform such results. So the model was still trained on uni-labeled task, but then as we get probabilities of being in a class as output, we will be able to assign multiple classes depending on a threshold.
To fix this threshold, we have annotated 200 articles randomly sampled in our test dataset. To make this annotations, we have build a streamlit platform which has helped us to annotate faster.



Data annotation

« Alvaro Gonzalez n'est pas raciste ». Au lendemain de sa victoire historique au Parc des Princes (0-1), l'Olympique de Marseille a tenu à défendre Alvaro Gonzalez, accusé de racisme par Neymar. « Alvaro Gonzalez n'est pas raciste, il nous l'a démontré par son comportement au quotidien depuis son arrivée au club, ainsi qu'en ont déjà témoigné ses coéquipiers », écrit le club dans un communiqué.Sans démentir formellement les accusations de racisme, l'Olympique de Marseille explique se tenir « à la disposition de la commission de discipline pour coopérer pleinement à l'enquête sur l'ensemble des évènements qui ont émaillé cette rencontre, et les 24 heures qui les ont précédés ». Le club condamne « la diffusion des numéros de téléphone privés d'Alvaro Gonzalez et de ses proches cette nuit sur des médias et réseaux sociaux brésiliens, donnant lieu depuis à un harcèlement de chaque instant, consistant notamment en des menaces de mort ».La commission de discipline tranchera ce mercredi sur les sanctions des joueurs ayant reçu un carton rouge : Neymar, Kurzawa et Paredes côté parisien, Benedetto et Amavi côté olympien. Elle pourrait également ouvrir une enquête sur les accusations de racisme de Neymar envers Alvaro Gonzalez. Mais lundi soir le diffuseur du match, Téléfoot la chaine, a expliqué n'avoir relevé aucune image ou son pouvant attester des accusations de Neymar. Celui-ci a d'ailleurs tenté de calmer le jeu ce mardi matin, tout en maintenant ses accusations.

sport

Figure 9: Our data-annotation platform

Then once the annotation done, we have computed for this 200 articles, the precision, recall and f1 score for each of the classes considered. Then after having performed a weighted average, we have obtained the final scores which have allow us to fix the optimum threshold.

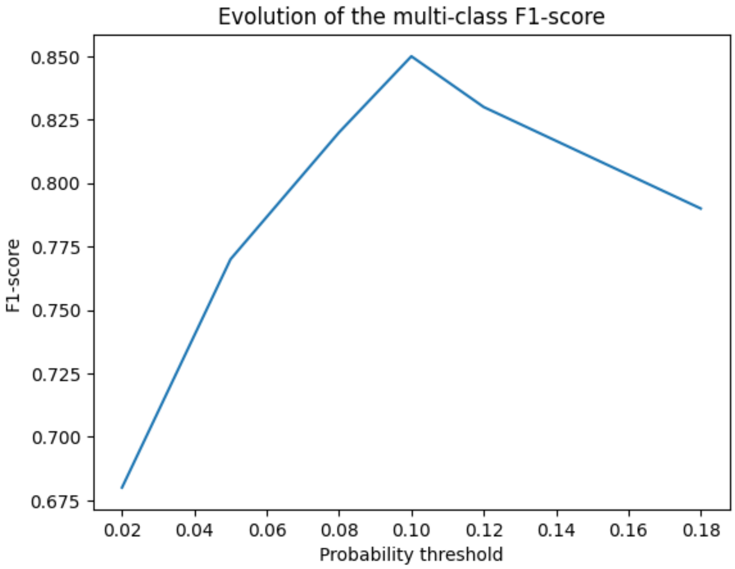


Figure 10: Graph to fix the probability threshold

4.5 Evaluation & Results

To compare the models, we have firstly extract the following seven categories from the newspaper ”20 Minutes” :

Category	Key words
Planet	Climate, environment, temperature, warming, nature
Sport	soccer, rugby, competition, match, score, cup
Politics	election, vote, referendum, parliament, minister
Economy	money, inflation, growth, debt, euro, tax, cost
Technology	computer, phone, server, cloud, screen, nanotechnology
minor news item	local
Entertainment	Stars, celebrities, cinema, theater

Then we have sampled 10 000 articles of each categories in the period 2006-2015 in order to built our train dataset, and 1000 articles of each dataset in the period 2015-2017 for the test dataset.

Then based on that, each model has been trained and tested on these datasets. We obtained the following results :

Model	Accuracy
Doc2Vec + GMM	49.7%
Lbl2Vec	74.8%
Camembert	88.1%
Combined Camembert	92.0%

We can thus observed that the Fine-tuned Camembert outperform the other methods. So we have kept this type of model for our next experiments.
The next step was to increase the number of categories to be more accurate and to perform multi-label classification to be closer to the reality of the distribution of subjects in an article. So we added the four following categories:

Category	Key words
Health	Drug, disease, doctor, vaccine
Justice	Conviction, judge, lawyer
Society	Population, demonstration
world	World, Country, war, USA

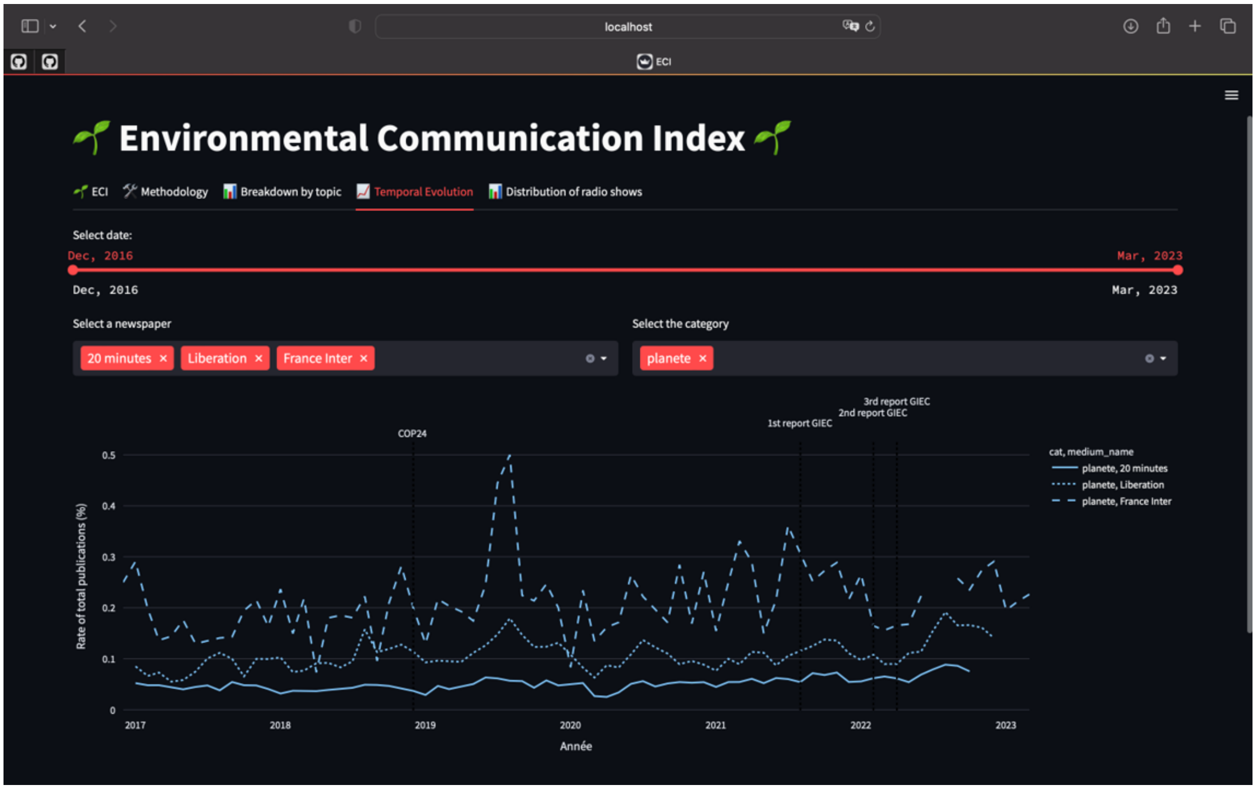
So we have fine-tuned a Camembert on the training dataset and then made predictions on the test dataset for this model and a zero-shot model as explained in the previous part. After having found the best threshold value, we obtained the following results :

Model	Accuracy	Recall	F1 score
Zero-shot	50.0%	67.0%	48.0%
Fine-tuned	84.0%	89.0%	85.0%

So the fine-tuned model showed the best results then we kept it for our predictions presented on our dashboard.

5 Our dashboard

Now that all the bricks of our project have been built, we decided to develop a small web application to showcase our results. The objective is manifold: The application is a way for us to share our work with the world, possibly enabling another team to take over. In addition, this allowed us to validate our previous work and helped us extract insights. In this application, we show the current state of environmental topics in the French media, to see the evolution with previous years.
We used streamlit, a library that Jean and Arnaud proposed. Very easy to use, it allows to deploy small platforms very quickly without worrying about a front-end and a back-end, and offers very fast deployment solutions.
The great advantage of these platforms is that they are interactive. Thus, on our platform, the user can also play with different parameters to see only the data he is looking for.



Select date:

Dec, 2016 — Mar, 2023

Select a newspaper: 20 minutes × Liberation × France Inter ×

Select the category: planete ×

Figure 11: Our dashboard: Temporal evolution

The dashboard is available online at this address: `https://martinlanchon-eci.streamlit.app`
You can see here that the user can choose the time interval they want, the newspapers or radio programmes they want, and focus the study on one type of theme to see how much it is mentioned. In addition, on this platform, we have added some markers that we think are important, and that can play a role in the share of topic in the media. When looking at the environment, we have added key events: COP 24 and the release of IPCC reports. In addition, there are several tabs with a reminder of the methodology followed, a zoom on the distribution of subjects, on the temporal evolution, and a zoom on the radio programmes, which have a rather particular format.

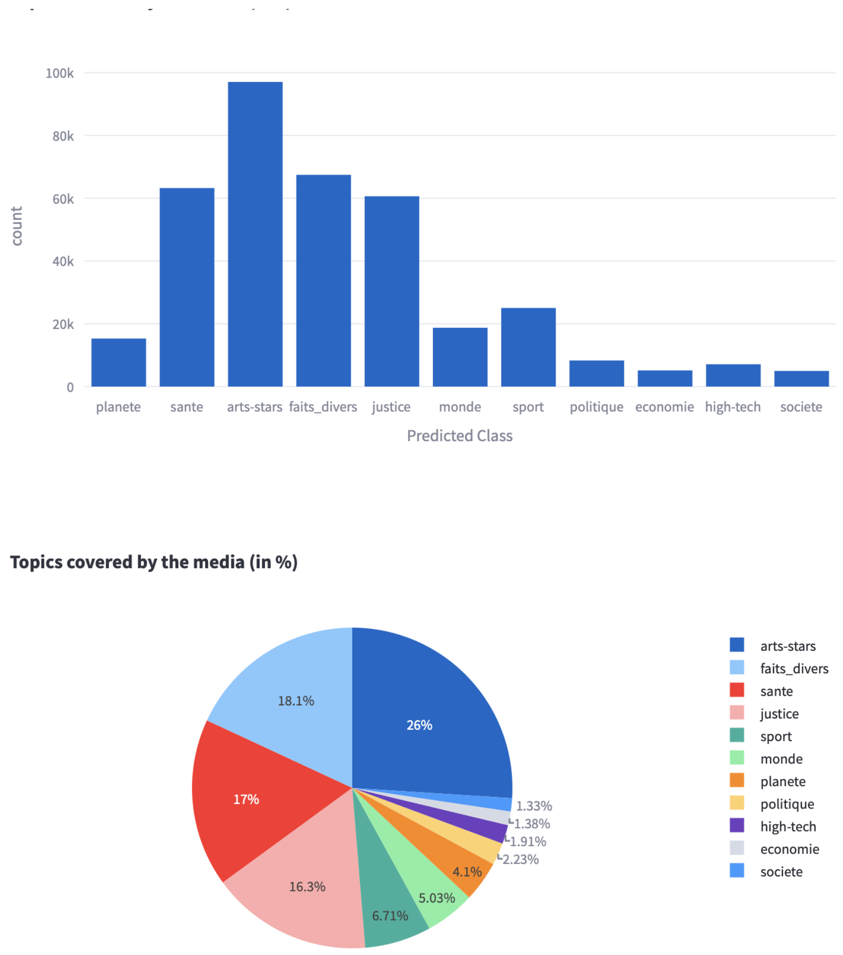


Figure 12: Our dashboard: Breakdown by topic

6 Conclusion

In this project, we were able to transcribe a few hundred radio broadcasts, retrieve millions of press articles, and classify them by topic. This allowed us to make a small visualisation platform to showcase these results and the evolution of environmental topics. All the code of our models can be found on our git, with the last versions in the main branch, and all our tests and work in the other branches: `https://github.com/AlexandrePradeilles/ECI`.

This subject was extremely formative for us because we learned about many state-of-the-art models in an extensive literature search. We were able to develop our own method or use pre-trained tools. We were trained on many of the key data science skills, from scraping, call API, NLP, cloud and visualisation, which made this project extremely comprehensive. We really enjoyed the opportunity to see many different tools and to do a project ”end-to-end”, from start to finish. Dealing with environmental issues that affect us was even more motivating.

Today, we consider that we can trust the results given to the platform, the classification results being good. However, an important limitation of the current tool is that we do not know how precisely these topics are addressed. This can range from a simple comment on the weather, to a real concrete environmental topic, to green-washing... This is why we think that this subject can be continued and pursued, to improve the models used if possible, and above all to study the way in which these subjects are raised. In addition, we would like to study more precisely the classification on radio, which has a somewhat different format, with many topics covered in one programme. Are 5-minute classifications really the best solution? With more time and resources, we would have continued the project in this direction.

7 Acknowledgments

We would like to warmly thank Eleven Strategy for helping us to carry out this project in the best possible conditions. A big thank you to Pietro for his trust and his wise advice, especially for the final presentation. We would also like to thank Arnaud and Jean, who supervised us perfectly, understood and accepted our sometimes complicated schedule, and adapted to all our constraints. We had a great deal of trust and freedom which was very much appreciated in the group, and your advice was always very valuable and saved us a lot of time. You were always available to answer our questions, and you always had solutions to propose in all situations. We also thank Kaniav Kamary, professor at CentraleSupélec, for her guidance and advice.

References

[1] Preethi Jyothi. Microsoft conference. `https://www.microsoft.com/en-us/research/video/automatic-speech-recognition-overview/`, 2017. 5

[2] M.R. Bouadjenek and N. D. Huynh. Automatic speech recognition using ctc. `https://keras.io/examples/audio/ctc_asr/`, 2021. 5, 11

[3] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural 'networks. volume 2006, pages 369–376, 01 2006. 7

[4] Picovoice. Speech-to-text benchmark. `https://github.com/Picovoice/speech-to-text-benchmark`, 2022. 8, 11

[5] Bryanpark. French-single-speaker-speech-dataset. `https://www.kaggle.com/datasets/bryanpark/french-single-speaker-speech-dataset`. 10

[6] Ilias Papastratis. Speech recognition: a review of the different deep learning approaches. *https://theaisummer.com/*, 2021. 11

[7] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach, 2019. 14

[8] Tim Schopf, Daniel Braun, and Florian Matthes. Lbl2vec: An embedding-based approach for unsupervised document retrieval on predefined topics. In *Proceedings of the 17th International Conference on Web Information Systems and Technologies*. SCITEPRESS - Science and Technology Publications, 2021. 15

[9] Louis Martin, Benjamin Muller, Pedro Javier Ortiz Suá rez, Yoann Dupont, Laurent Romary, Éric de la Clergerie, Djamé Seddah, and Benoît Sagot. CamemBERT: a tasty french language model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2020. 16

[10] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017. 16

[11] Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. Finetuned language models are zero-shot learners, 2022. 17