UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH

Escola Tècnica Superior d'Enginyeria
de Telecomunicació de Barcelona

Age Prediction by voice using Deep Learning

A Master's Thesis
Submitted to the Faculty of the
Barcelona Technical School of Telecommunication Engineering
Universitat Politècnica de Catalunya
by

David Linde Martínez

In partial fulfilment
of the requirements for the degree of
MASTER IN TELECOMMUNICATIONS ENGINEERING

Advisor: Francisco Javier Hernando Pericas

Barcelona, January 2023

Abstract

One of the main topics in artificial intelligence is the speech characterization. Moreover, it is a field of study with minimal scope when the Catalan language is involved in. In this project, we try to perform an age classification by decades firstly in the Catalan CommonVoice Dataset [1] and then add the Spanish Dataset and English Dataset to have more data. To reach our purpose Deep Learning techniques are used to implement the classifier. The most common backbones are used such as Resnet and VGG. Furthermore, we use an attention encoder to encode the Mel-Spectrogram features. In contrast to statistical pooling methods like average pooling, Attention Pooling layers and various Attention Mechanisms are used in all backbones to perform pooling and reduce the dimensionality of the feature vector derived from the Front-End architecture. In this study, we will compare two different models, the first with an AM-Softmax in the final layer and the other with an AM-Softmax combined with Ordinal Regression.

Acknowledgments

I want to give my thanks to the team of voice research that is involved and also related to my thesis, in particular to my tutor Francisco Javier Hernando Pericas for guiding me in this project and for all the teachings during my Bachelor's and Master's Degree. I would also give thanks to my family who is always supporting me and giving me the opportunity to study whatever I want.

Revision history and approval record

Revision	Date	Purpose
0	05/01/2023	Document creation
1	14/01/2023	Document revision
2	18/01/2023	Document revision

Written by:		Reviewed and approved by:	
Date	05/01/2023	Date	18/01/2023
Name	David Linde Martínez	Name	Francisco Javier Hernando Pericas
Position	Project Author	Position	Project Supervisor

Table of contents

List of Figures

List of Tables

1. Introduction

1.1. Motivation

Many of the most common voice assistants are widely used by everyone for various purposes in the age of AI and new technologies. To continue developing and improving these systems we think about the importance of the age of the speaker. If we are able to know the age of the speaker, we can refer to them more precisely.

Furthermore, It is evident that only a few languages are supported for interaction with technology, and it is clear that not all languages are being developed at the same rate. This is particularly true for Catalan, which is not a priority for many companies and therefore lacks the resources and material needed for the development of AI. Therefore, it is important for research to focus on creating a solid foundation of knowledge and resources that can be used and built upon by other developers and companies. Without efforts to develop AI in Catalan, it is unlikely that anyone else will take on this task. Ensuring that everyone can use their native language with new technologies, there are numerous open-source initiatives that aim to support this goal, such as CommonVoice, which is a global, open-source voice database containing a wide range of audio clips in different languages. This type of dataset can be used to develop various speech technology applications.

In order to achieve the best performance, we will use the state of the art in deep learning technologies such as Resnet, AM-Softmax, and Attention mechanisms.

1.2. Objectives

The main objective of this study is to find a model to help us to classify the age of a speaker in a rank of a decade. It will be a good performance if the precision is better than humans which will be compared at the end of the document. Moreover, it's necessary that this project could help other people in the future such as baseline to improve it or other related fields of study.

To be more concrete the main objectives are

1. Visualize the data of the dataset and clean it for our project.
2. Classify age in rank ages by analyzing the spectrogram. We will have 6 age ranges: twenties, thirties, forties, fifties, sixties, and seventies.

3. In order to improve the baseline model, combine AM-Softmax loss with ordinal regression loss.
4. Feature visualization to compare different models

1.3. Outline of the thesis

Has the following structure:

- Chapter 2 gives the state of the art in the prediction of age with the voice. Gives an overview of techniques of the past and also the ones that are used nowadays. Finally, there's an explanation of the methodology that is used in the thesis.
- Chapter 3 explains how it implemented the methodology that is explained in chapter 2.
- Chapter 4 presents the different experiments and also all the results and feature visualization.
- Chapter 5 there are the final conclusions, comparison with human performance, and also future work.

2. State of the art

The study related to the processing of speech includes various fields and tasks such as speaker recognition, emotion recognition, and gender classification. In this case, we are seeking to classify age from speech signals. To achieve this, we will use the most advanced tools available, specifically machine learning and deep learning techniques.

Nowadays, Deep Learning techniques that process and learn from speech data use Mel-Spectrograms as input, which are two-dimensional representations of voice signals with one dimension representing time and the other representing frequency transformed into the Mel scale which is created due to human perception. These Mel-Spectrograms are then processed by a feature extractor and a model utilizing Attention mechanisms, before being classified through the use of pooling functions or similar techniques.

This chapter provides background information on the conventional methods and current approaches in this field, as well as the specific functions and operations that will be used in this work. Finally, the architecture that will be utilized in this thesis is presented.

2.1. Back to the past

Several methods have been proposed to address the problem of age prediction from speech signals. These approaches are often hybrid models that consist of two stages, focused on finding the optimal features and designing a classifier model. Many of these methods rely on Dense Neural Networks (DNNs) and use Hidden Markov Models to output probabilities for training the DNNs.

These models perform well when there is a small amount of training data available. I-Vectors have also been commonly used as optimal feature vectors for speech tasks, and Support Vector Regression has been used to create an age prediction model. Other approaches have been utilized such as Gaussian Mixture Models (GMMs) and Support Vector Machine (SVM) models through a score-level fusion model. Despite the various proposals for age prediction from speech signals, it remains challenging to extract optimal features and design high-performing classification models. Additionally, one of the main reasons for low results in age classification using acoustic features is the similarity of frequency-related acoustic features across different age groups.

It is a challenging problem because, for humans, it is also difficult to predict the age of a speaker. For example, it is very difficult to distinguish between the voices of someone in their forties and someone in their fifties using spectrograms. If we compare the

spectrograms of someone saying the same sentence over a ten-year span, we would likely see that they are almost identical. This illustrates the challenges in distinguishing between the voices of people within a relatively close age range using acoustic features alone.

So it is not like other types of deep learning challenges in which humans are very good at some type of prediction or whatever. This makes the problem harder to solve.

2.2. Feature extraction

In this section we are going to focus on structures called End-to-End (E2E) which are deep learning approaches. When we are working with audio and voice in these types of structures, firstly, we need to extract some type of features to later give as input in the deep learning system. There are many different techniques for extracting features from data, and many algorithms that can be used to classify speakers' voices. Spectral features, such as Mel Frequency Cepstral Coefficients (MFCC), Linear Prediction Coefficients (LPC), and Line Spectral Frequencies (LSF), are commonly used for this purpose. Among these, the Mel/Log-Mel Spectrogram is the most widely used.

Mel-Spectrogram: The Mel-Spectrogram is a type of spectral representation of audio signals that are based on the human auditory system's perception of sound because the human perception of frequencies is non-linear, with better sensitivity to differences in lower frequencies compared to higher frequencies. This has been demonstrated in research. Is also used for music recognition, as well as for other tasks related to audio analysis.

The Mel-Spectrogram is created by applying a series of transformations to the power spectrum of an audio signal. First, the power spectrum is computed using a Fast Fourier Transform (FFT) on the overlapped window segments of the signal. Then, the power spectrum is mapped onto the Mel scale, which is a scale of frequencies that is based on the way that the human ear perceives different frequencies. The resulting Mel-Spectrogram is a two-dimensional representation of the audio signal, with time on the x-axis and frequency on the y-axis.

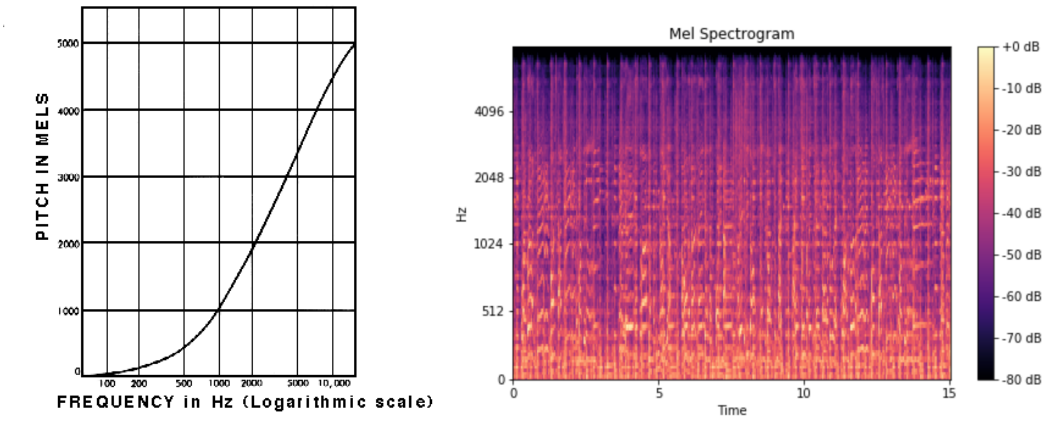

Figure 1: Mel Scale & Mel-Spectrogram

2.3. Most common front-ends: VGG and Resnet

Nowadays, Most advanced Deep Learning systems for speaker verification utilize speaker embedding extractors. These architectures often include a Front-End to extract features, the input of these embedding extractors is the Mel-Spectrogram.

In the context of speech processing, these features are typically designed to capture important characteristics of the speech signal that are relevant for the task at hand, such as the spectral content of the signal, the pitch and energy of the speaker's voice, and other relevant characteristics. A front-end feature extractor is an essential component of a speech processing system, as it plays a key role in the extraction of relevant and meaningful features from raw audio data.

The output of the Front-End goes into a pooling layer that converts variable-length input into fixed-length vectors, which are then processed by a feed-forward neural network and a softmax or sigmoid layer to classify the output into one of the available classes.

So, it is crucial for the performance of the extraction of features because they are input to other parts of the system.

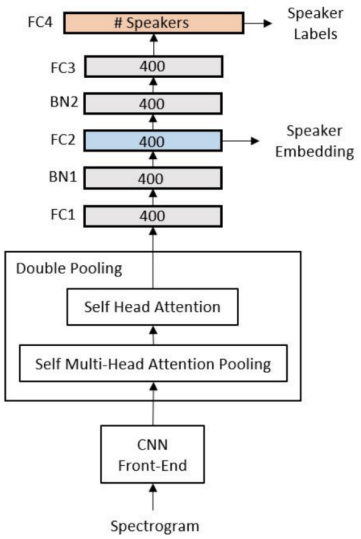

Figure 2: Structure of a Speaker recognition E2E system that includes a Front-End [2]

In order to extract the best features, the researchers use backbones which are a neural network architecture that serves as the foundation for a larger model. It is typically a pre-trained model that has already been trained on a large dataset and can be fine-tuned for a specific task by adding additional layers on top of it. The purpose of using a backbone in deep learning is to take advantage of the knowledge and features learned by the pre-trained model and use them as a starting point for training a new model. This can save time and resources, as the backbone has already learned many important features that are useful for a wide range of tasks. In this work, we are not using a pre-trained model, only the structure of the front-end.

There are several different types of backbones that are commonly used in deep learning, the most common are VGG [3] and Resnet [4]. Over the years in computer vision research, VGG has been the predominant front-end but this has changed in recent times due to Resnet outperforms VGG. Regarding speech processing, it is not very clear if Resnet outperforms VGG, this will be investigated in this study.

The VGG is a convolutional neural network composed of 19 layers with 3x3 convolutional filters with stride 1, max pool filters of shape 2x2, and stride 2. This network shows the importance of depth.

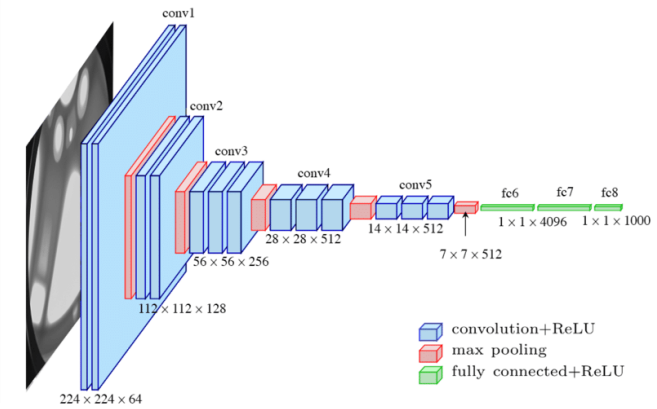

Figure 3: VGG16 structure. Figure extracted from the paper [3]

The Resnet is a convolutional neural network that is characterized by the use of residual blocks, which are designed to allow the network to learn residual functions concerning the input. This helps to alleviate the vanishing gradient problem, which can occur in very deep networks, and allows the ResNet to train very deep networks effectively.

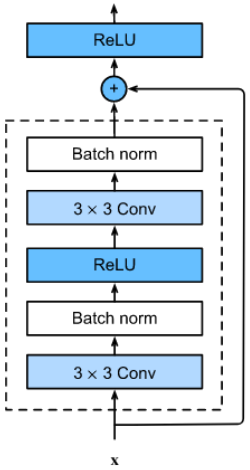

Figure 4: Main structure of a Resnet. Figure extracted from [7]

2.4. Self-Attention, and Multi-Head Attention

As I said, one of the main problems of deep learning structures is the vanishing gradient problem [5]. To reduce the difficulty to contextualize the long-term patterns in a sequence there's a state-of-the-art solution called attention.

Attention mechanisms are a way for a model to selectively focus on specific parts of its input when processing it. They allow the model to dynamically weigh different input features and determine which ones are most important for a given task.

In the context of natural language processing (NLP), attention mechanisms allow a model to focus on specific words or phrases in a sentence when making predictions or generating text. This is particularly useful for tasks such as machine translation, where the model needs to be able to understand the meaning of a word in the context of the entire sentence. Attention mechanisms can be implemented in various ways, but one common approach is to use a neural network to learn to weigh the importance of different input features based on the task at hand. The model then uses these weights to selectively focus on certain parts of the input when making predictions or generating output.

In an age classification task, the attention layer is likely to highlight the features that contain information relevant to determining the age of the speaker, which are likely to have the most significant influence on the classification. In attention mechanisms, we have the terms "query", "key", and "value" which refer to the inputs that are used to calculate the attention weights for each element in a sequence.

- "query" is a representation of the current state of the model and is used to determine which elements in the input sequence are most relevant for the task at hand.
- "key" is a representation of each element in the input sequence and is used to determine how closely it matches the current query.
- "value" is a representation of the importance of each element in the input sequence and is used to weigh the contribution of each element to the final output of the model.

Together, the query, key, and value are used to calculate the attention weights for each element in the input sequence, which are then used to determine which elements the model should focus on when processing the input. Once we have these concepts clear, we can compute several types of attention functions such as Additive Attention, Multiplicative and Dot Product Attention, and Scaled Dot Product Attention.

Self-Attention

Also known as "intra-attention," is a type of attention mechanism that allows a model to focus on different parts of its input when processing it. In self-attention, the model calculates attention weights for each element in the input sequence based on the relationship between the element and all the other elements in the sequence. These attention weights are used to weigh the contribution of each element to the final output of the model.

The (dot product) self-attention operation is as follows:

1. Compute key-query affinities:

$$e_{ij} = q_i^T K_j$$

2. Compute attention weights from affinities:

$$\alpha_{ij} = \frac{exp(e_{ij})}{\Sigma_j exp(e_{ij})}$$

3. Compute outputs as a weighted sum of values:

$$output_i = \sum_j \alpha_{ij} v_j$$

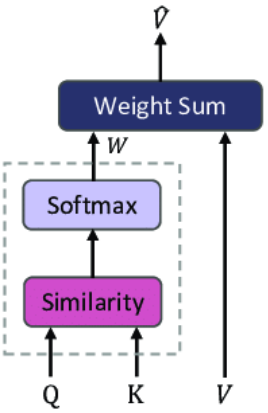

Figure 5: Self-Attention module, figure extracted from [9]

Self Multi-Head Attention

Self-multi head attention is a variant of self-attention, a type of attention mechanism that allows a model to focus on different parts of its input when processing it. In self-multi-head attention, the model calculates attention weights for each element in the input sequence using multiple "heads," each of which focuses on a different subset of the input elements.

The attention weights calculated by each head are then combined and used to weigh the contribution of each element to the final output of the model. This allows the model to attend to multiple different aspects of the input at the same time, improving its ability to understand the relationships between the elements in the input sequence.

So, if from the output of the front-end feature extractor we have $h = [h_1 h_2 \dots h_n]$ with $h_t \in R^D$ and if we consider a number K heads for the MHA pooling now we can define the state as $h_t = [h_{t1} h_{t2} \dots h_{tk}]$ where $h_{tj} \in R^{D/K}$. Thus, each feature vector is split into a size of D/K. Finally, the weights of each head alignment are defined as:

$$w_{tj} = \frac{exp(\frac{h_{tj}^T u_j}{\sqrt{d_h}})}{\Sigma_{l=1}^{N} exp(\frac{h_{tj}^T u_j}{\sqrt{d_h}})}$$

In the next image, we can see the structure of Self Multi-Head Attention, with several heads to attend different sets of tokens.

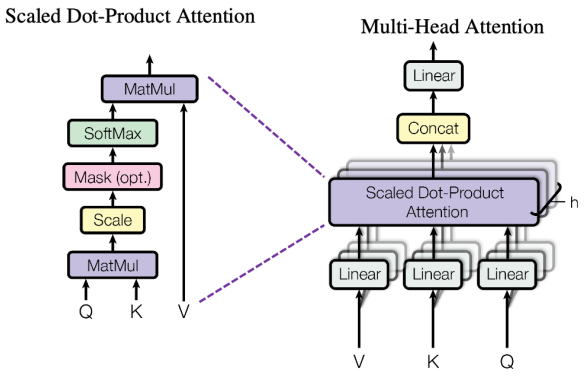

Figure 6: Self Multi-Head Attention module, figure extracted from [10]

2.5. Pooling and Double Multi-Head Attention

In deep learning, pooling is a type of sub-sampling operation that is applied to the output of a convolutional layer. The main purpose of pooling is to reduce the size of the input tensor, which helps to reduce the number of parameters and computations in the model, and makes the model more efficient and faster to train.

There are several types of pooling operations that are commonly used, including max pooling and average pooling. Pooling methods also could be attention mechanisms such as Self MHA. The main issue with Self MHA pooling is that it assumes that all heads have equal relevance. The resulting context vector is created by combining all of the head context vectors and is used as input for the subsequent dense layers. In contrast, Double MHA[2] does not make this assumption, allowing each utterance context vector to be computed as a unique linear combination of head context vectors.

2.6. AM-Softmax

AM-Softmax (Additive Margin Softmax) [6] is a loss function that is used in deep learning for training models for facial recognition tasks. It is a variant of the widely-used Softmax loss function, which is used to train models to predict class probabilities in classification tasks.

In AM-Softmax, the loss function is modified to include an additive margin term designed to increase the separation between different classes in feature space. This helps improve the model's performance by making it more difficult for the model to confuse different classes.

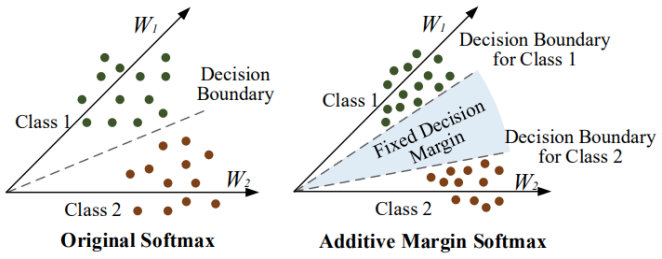

Figure 7: Comparison between Softmax and AM-Softmax, figure extracted from [6]

This loss function is typically used in combination with a convolutional neural network (CNN) for facial recognition tasks, where it has been shown to improve the performance of the model compared to using the standard Softmax loss function.

Furthermore, it is not only used in face recognition, it is also used in speaker recognition [2], and in this study will be used for age classification. It will help us to increase the separation between the different ranks of ages.

2.7. Ordinal Regression

Is a type of statistical analysis used for predicting an ordinal variable, or a variable that has a limited number of ordered categories. It can be considered an intermediate problem between regression and classification.

Ordinal regression models are used to predict the category or level of the ordinal variable based on one or more independent variables. They are similar to linear regression models, but with some modifications made to handle the ordinal nature of the dependent variable.

For example, a survey may ask respondents to rate their satisfaction into categories $A < B < C < D < E$ where the category A is very unsatisfied and the category E very satisfied. In this case, the variable "satisfaction" is ordinal, as there are only five possible categories. The challenge in utilizing a standard classifier to address this type of issue is that the model will treat the mistake of incorrectly identifying an A as a D as equivalent to identifying A as a B, which is clearly not accurate since the discrepancy between A and D is much greater than between A and B.

Therefore, this same problem is also applicable when classifying age in decades because they are ordered.

3. Project development

In this chapter we will focus on the methodology used in this project. Firstly, there is an explanation of the dataset. Secondly, how we manage the data that the dataset provides us and also why and how we extend the data available. Then, there's an explanation of the basic architecture obtained from [2]. In the following, we will describe two modifications of this architecture that involve altering the front-end or feature extractor.

3.1. Common Voice Dataset

The Common Voice dataset [1], provided by Mozilla, contains a wealth of recorded voices in numerous languages, including Catalan. Not only do we have a supervised Catalan dataset, but this set has also recently been updated, enhancing the Catalan Common Voice dataset one of the driving factors behind this project. The version that we are using in this project is v11.0 released in September of 2022 which contains 1.701 validated hours and 30.225 speakers.

While they have been validated, not all of the samples have all the required information. As a result, we may not have labels for age tasks. Additionally, in order to have a well-balanced dataset that allows the model to learn effectively, we are also undersampling the dataset and removing some samples using a strategy that is specific to each task.

Moreover, to avoid overfitting the users, there's a limitation in the number of recordings that a user provides for the training, validation, and test. This has been done because there are some users with a massive amount of recordings, for instance, there is a person with more than 90.000. This produces a significant bias at the training time so it is necessary to take it into care.

For the partition of the dataset, a stratified split has been done. 80% of the data goes to training, 10% goes to validation and the 10% remaining is for testing.

3.2. Data preparation

To have the best possible dataset for training, we have applied some different kinds of filters. Firstly, we have removed the audio samples that have less than 100 frames of Mel-Spectrogram. This is around a second of time recording.

We have done a plot to visualize the distribution of the number of frames in the dataset.

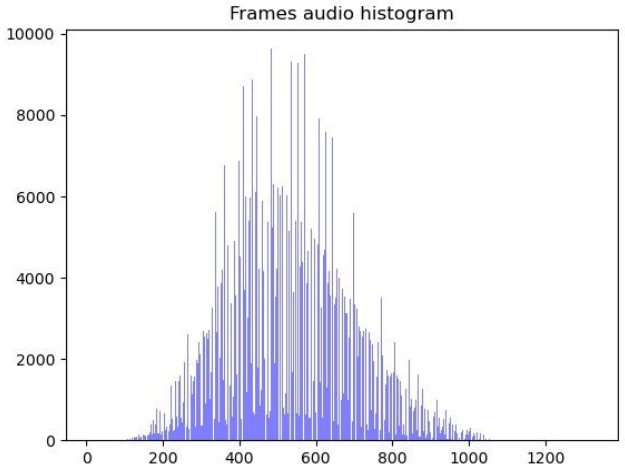

Figure 8: Histogram of the number of frames in the dataset

As we can see in the plot, there's very little data below 100 frames.

Furthermore, as I said in the previous section, in the dataset there is a very huge class imbalance, in the next image we can see the distribution of age classes for the Catalan dataset.

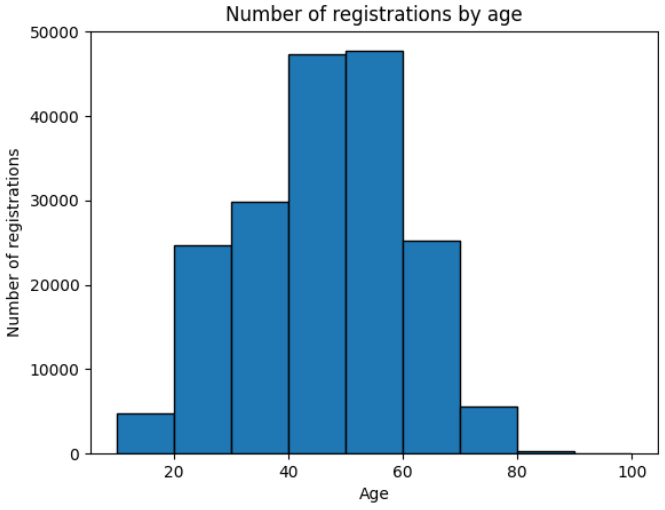

Figure 9: Histogram of age class (Catalan) v1

The rank ages in this dataset go from teens to nineties, the plot shows us that the distribution is not equal so in order to achieve better results we have deleted the teens class and also we have mixed the classes eighties and nineties with seventies to have more data in old ages.

Once we have done these modifications, the new distribution is the next one:

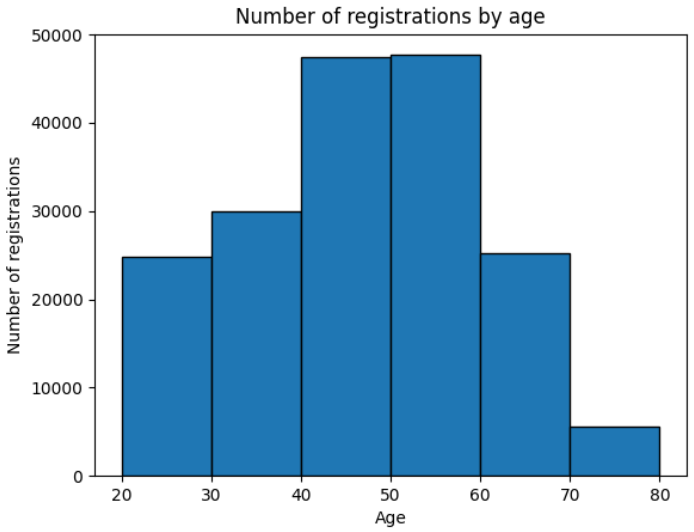

Figure 10: Histogram of age class (Catalan) v2

Now is a better distribution, although the class imbalance problem continues, so to improve the performance we have limited the classes forties and fifties to 30.000 samples. The other classes have enough registrations except for the seventies class which only has around 6.000 recordings.

Finally, for the Catalan dataset, we have 116.742 samples in training, 14.947 in validation, and 14.241 in testing. This is not a huge amount of data to reach a good model so this is going to be reflexed in the results section.

3.3. Data augmentation

To improve the results in age classification, we are going to augment the data available for training, validation, and testing. From the Catalan dataset, is not possible to get more recordings so we will add more languages to the dataset. After looking at the webpage of CommonVoice, the languages with the most samples are English and Spanish.

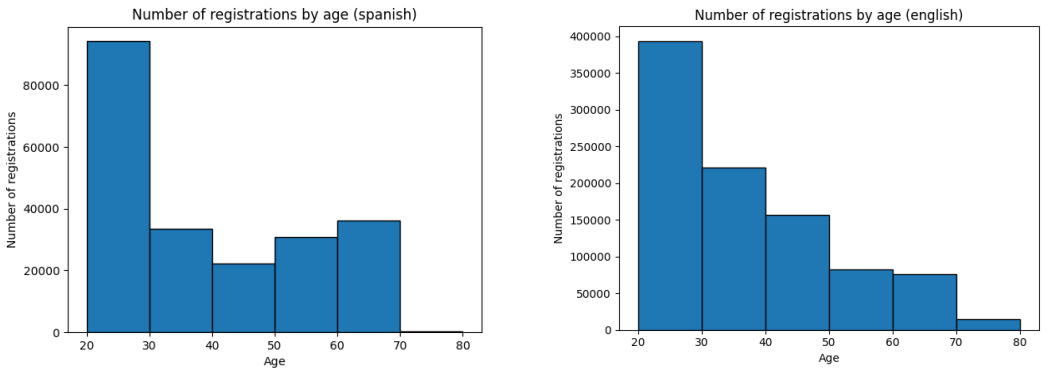

Figure 11: Histogram of age class (Spanish & English)

In the two datasets, also there is a class imbalance. The twenties class is predominant and the seventies class is the one with less number of registrations. At least, in the English dataset, the number of registrations for the other classes is very high and for the seventies is 12992. The Spanish dataset, it's a more balanced dataset but we have the sample problem as the Catalan dataset, the seventies class has a very low number of recordings.

To mix the three datasets we have applied the same techniques, as the ones explained before.

The distribution class of the final dataset which obtains the best results is the following one.

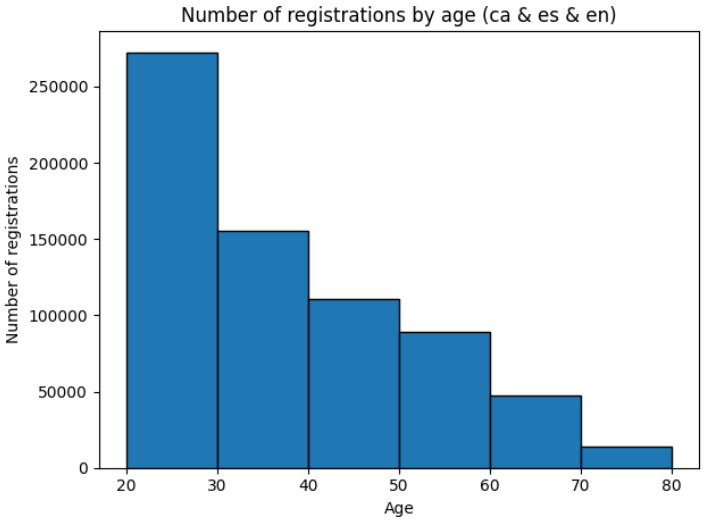

Figure 12: Histogram of age class (Catalan & Spanish & English)

Now, we have enough data in all the classes except for the seventies we have 14.161 though anyway we have improved the quantity of data in this class and it will help in the training. To have the most possible balanced dataset, a limitation in the number of registrations has been done. The maximum number of registrations for any class is 50.000. This affects all the classes except for the sixties (46.000 registrations) and seventies.

Finally, for the mixed dataset, we have 177.256 samples in training, 22.103 in validation, and 22.215 in testing.

3.4. Architectures

This chapter is about the presentation of the two architectures that we are going to use in the experiments and also compare them in performance. Firstly, we present an architecture only using AM-Softmax, which is the simplest. Then, there is the explanation of the second architecture which combines AM-Softmax with Ordinal Regression.

3.4.1. Architecture 1: Only using AM-Softmax

In this section, there's an explanation of the first model of age classification. That's the basic model.

It consists of a Front-End, which could be a VGG4L or a Resnet, to extract the features of the Mel-Spectogram followed by a Double MHA Pooling block to reduce the dimensionality of the feature vector. Then a fully connected layer with ReLU function and finally the AM-Softmax to classify and also include an additive margin term designed to increase the separation between different classes in feature space.

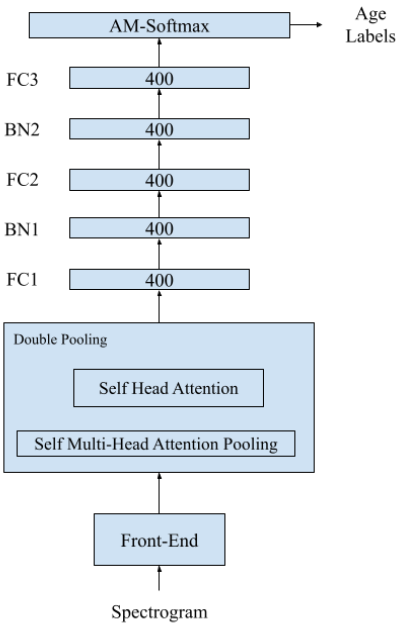

Figure 13: Architecture 1: AM-Softmax

The AM-Softmax has two outputs, one is for the predictions of the model and the other one is to compute the loss of the system. The loss output is the prediction with an addition of the margin term multiplied by a scaling factor. Then, to compute the loss of the global system there's the application of the weighted Cross-Entropy Loss function to these second output of the AM-Softmax.

We are using a weighted Cross-Entropy Loss function because we have an imbalanced dataset, so we need to give different weights and importance to each class.

$$classWeights = \frac{Nsamples}{Nclasses \bullet Noccurences(class)}$$

In the end, we adjusted the neural network weights through backpropagation and repeated this process over several epochs

3.4.2. Architecture 2: AM-Softmax combined with Ordinal Regression

In this section, we treated the age classification task as a regression problem, although regression methods tend to perform worse than classification methods.

To address this, we modified the regression loss function specifically for the Age Classification task. The need to perform these changes in the loss function comes from that the age classes are ordered. For instance, if a speaker is in the range of thirties and a classification model classifies the speaker as twenties is better than classifying the speaker as sixties because it has a smaller error.

To not lose the powerful class separation of AM-Softmax for the classification system, our new model is a combination of AM-Softmax and Ordinal Regression. This is done with the sum of the two losses. So, when we do backpropagation we will take into account the classification and ordinal regression problem. For the Ordinal Regression, we need to do One-hot encoding to our class labels to be able to calculate the loss between the prediction and the One-hot class.

Ordinal Regression target vector	
Decade	**One-hot encoding**
Twenties	[1,0,0,0,0,0]
Thirties	[0,1,0,0,0,0]
Forties	[0,0,1,0,0,0]
Fifties	[0,0,0,1,0,0]
Sixties	[0,0,0,0,1,0]
Seventies	[0,0,0,0,0,1]

Table 1: One-hot encoding

So, to get an input for the Ordinal Regression loss, we perform a Softmax function to the prediction output of the AM-Softmax. Because the prediction output of the AM-Softmax doesn't give the output vector with a range [0,1]. In this way, we will have a vector that has all their components between [0,1] and its component will represent the probability of being in each class.

With these types of input, we are able to calculate the distance between the prediction and the perfect prediction which is one-hot encoding. Furthermore, to have into account that the classes are ordered, we can penalize the wrong components of the vector which have a probability bigger than 0. With this method, the model will learn that the classes are ordered and also that an error of 3 decades is worst than 1 decade. This will provide us more diagonal matrix.

The loss function for a batch of predictions is the following one. Firstly, we compute a modified prediction that represents the prediction multiplied by the penalization to each component.

$$modifiedPrediction = prediction_i \bullet abs(target - i + 1)^2$$

So, depending on the distance of the component to the target, we will penalize differently. The squared elevation is due to is easier to perform the backpropagation and descent of the gradient when we are training.

Then, we perform the mean squared error with all the $modifiedPrediction$ and the one-hot encoding for every target.

$$RegressionLoss = \sum_{j=0}^{N-1} \frac{1}{N} (modifiedPrediction_j - target_j)^2$$

Now we have the ordinal regression loss and it's time to do the sum with the weighted Cross-Entropy Loss that we obtain from AM-Softmax. To perform this addition, we will create a parameter α that will ponderate each loss depending on its importance. This parameter will be a trainable parameter, so when the backpropagation is performing will be re-adjusting α. The parameter is initialized at 0.5 to give the same importance to the two losses at the beginning of the training.

$$loss = (1 - \alpha) \bullet AMSoftmaxLoss + \alpha \bullet RegressionLoss$$

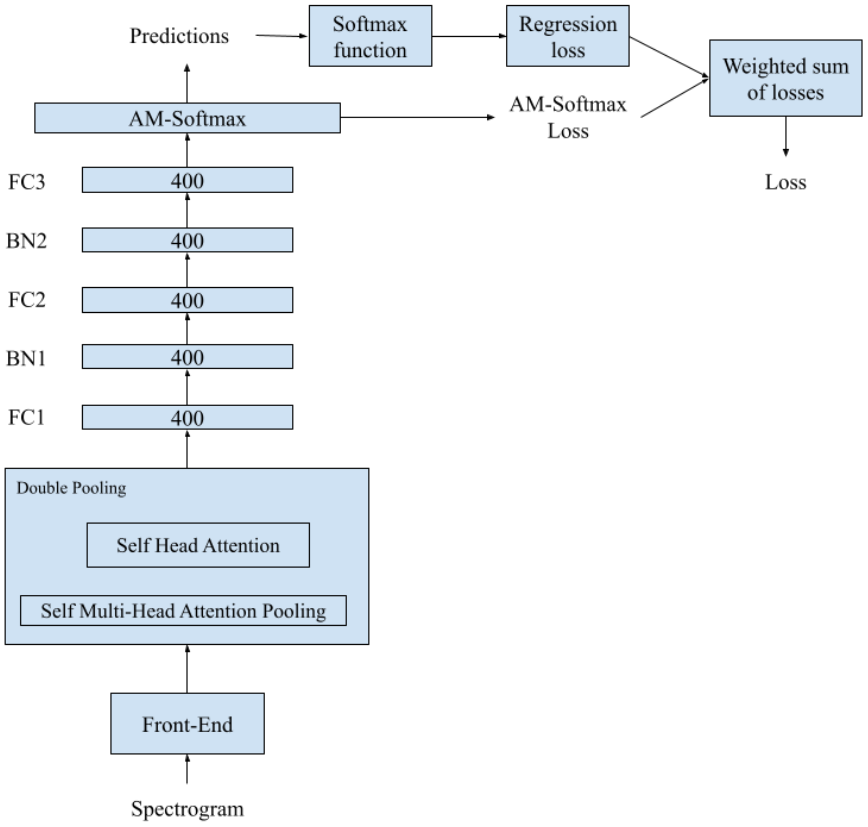

Figure 14: Architecture 2: AM-Softmax combined with Ordinal regression

3.5. Performance metrics

To evaluate our system, we need some type of metrics. In this project, we evaluate our system using three metrics, F-score (macro), Modified F-score (macro), and MSE. The most important metric is the F-score because is used to evaluate the accuracy of a classifier. It is defined as the harmonic mean of precision and recall.

$$Fscore\,(macro) \;=\; 2 \bullet (P \bullet R)/(P + R) \;=\; \frac{TP}{TP+\frac{1}{2}(FP+FN)}$$

There are different types of F-score metrics, in our case, we use the called Macro F-score because our dataset is imbalanced, and all classes are equally crucial to us.

Moreover, as is explained in the first chapter of this report, making a mistake of a decade is not a huge error because the training labels of the Common Voice dataset are on decades. So, for instance, a person who is 29 years old, will choose the twenties labels as the same as a person who is 21 years old. This produces similarities between the

neighboring classes. Because of this, we have implemented a Modified F-score that gives a good classification in neighboring classes.

For example, if a person is in their thirties and we classify him in the adjacent classes of twenties or forties, with the Modified F-score we give as valid this classification. This will produce a higher score.

Finally, we also use the mean squared error (MSE) which will compute the distance in a number of decades for every prediction and the target. Then compute the mean of all the distances. Our purpose is to minimize this error.

$$MSE = \sum_{j=0}^{N-1} \frac{1}{N} (x_j - y_j)^2 \; ; \; x_j(pred) \; ; y_j(target)$$

To conclude, in order to save the best model when we are training, we save the model which reaches the best F-score (macro) in the validation process.

4. Results

This chapter is about to show and discuss the experiments of the results in age classification. In every experiment, we will present the different types of metrics and also the confusion matrix.

First, we only use the Catalan dataset to make the comparison in performance between VGG4L and Resnet, this comparison will decide which front-end is used in the following experiments. Also, we perform the comparison with the two architectures. Then, with the data-augmented dataset, we do it again the architecture comparison. Finally, we have the feature visualization and centroid distances of these two models.

4.1. Results with Catalan dataset

This section is about the comparison of VGG4L with Resnet and also the comparison of the two models presented in the chapter before only applying the Catalan dataset. We will see the behavior for each model with this lack of data with the different types of metrics.

4.1.1. Comparison between VGG4L and Resnet

Firstly, we present the used architecture of the front-ends. Because there are some particularities.

VGG4L: Consists of a 4-layer VGG front-end followed by a pooling layer to transform variable-length input into fixed-length class vectors. The VGG4L comprises 4 blocks of 2D Convolutional Neural Layers with ReLu activation and MaxPooling layers in between each block. It works as a feature extractor.

The output image is reshaped into a tensor with the shape:
$(batch\ size,\ length(audio)/16,\ Mel\ features\ \bullet\ feature\ dimension)$

Finally, the tensor goes throw the Double MHA Pooling layer, fully connected layer and, classified by an AM-Softmax.

Resnet: Consists of three Resnet blocks and each one of these blocks contains two convolutional layers. Finally, the output is passed throw the Double MHA Pooling block, fully connected layer, and classified by an AM-Softmax.

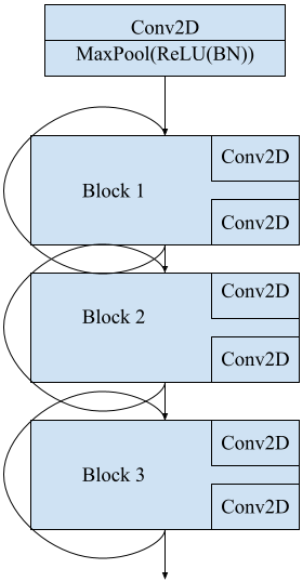

Figure 15: Resnet basic module

The difference in performance between the two front-ends has been done only in the Catalan dataset because the available data in this dataset is enough to decide which front-end is better.

Comparison between VGG4L and Resnet			
	F-score	**Modified F-score**	**MSE**
VGG4L	0.231	0.610	1.312
Resnet	0.250	0.639	1.224

Table 2: Comparison between VGG4L and Resnet

As we can see in the table, it is crystal-clear that Resnet has better performance than VGG4L in age classification. It's better in the three metrics and also in the training section, the overfitting of the VGG4L was substantially higher, but this does not help us if we want to add more modules to the system. So, Resnet is the chosen front-end for the following experiments.

4.1.2. Comparison between the two architectures

The result, of AM-Softmax is the same one because is the same model, as the Resnet results in the previous section.

Firstly, we are going to present the confusion matrices on the test set which are normalized over the true (rows), and then the table with the different metrics.

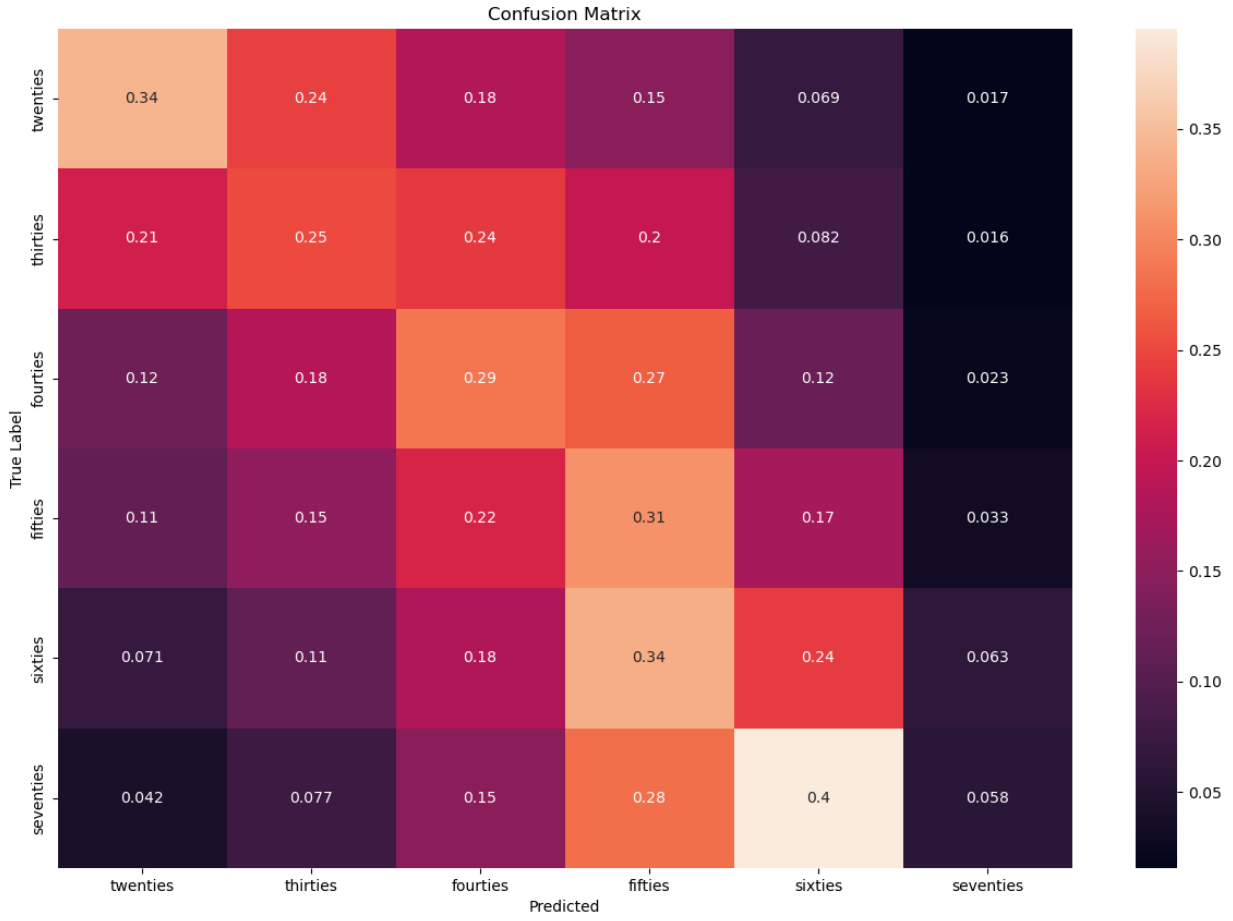

Figure 16: Confusion Matrix of only AM-Softmax (Catalan)

In the above plot, which is the architecture of only using AM-Softmax we can see that the matrix tries to be a diagonal matrix. The classes with better performance are the twenties and the fifties, this is due to the 'twenties' class only having one neighbor class so it's less probable to lose classifications in the near classes. The 'fifties' class is the class that has more data so this has helped when training. The 'seventies' class obtains a very bad result because of the lack of data. Although the classification in all classes is not quite good, we can see in the confusion matrix that if a recording does not classify in his class, it classifies in the neighbor class so is not a big error.

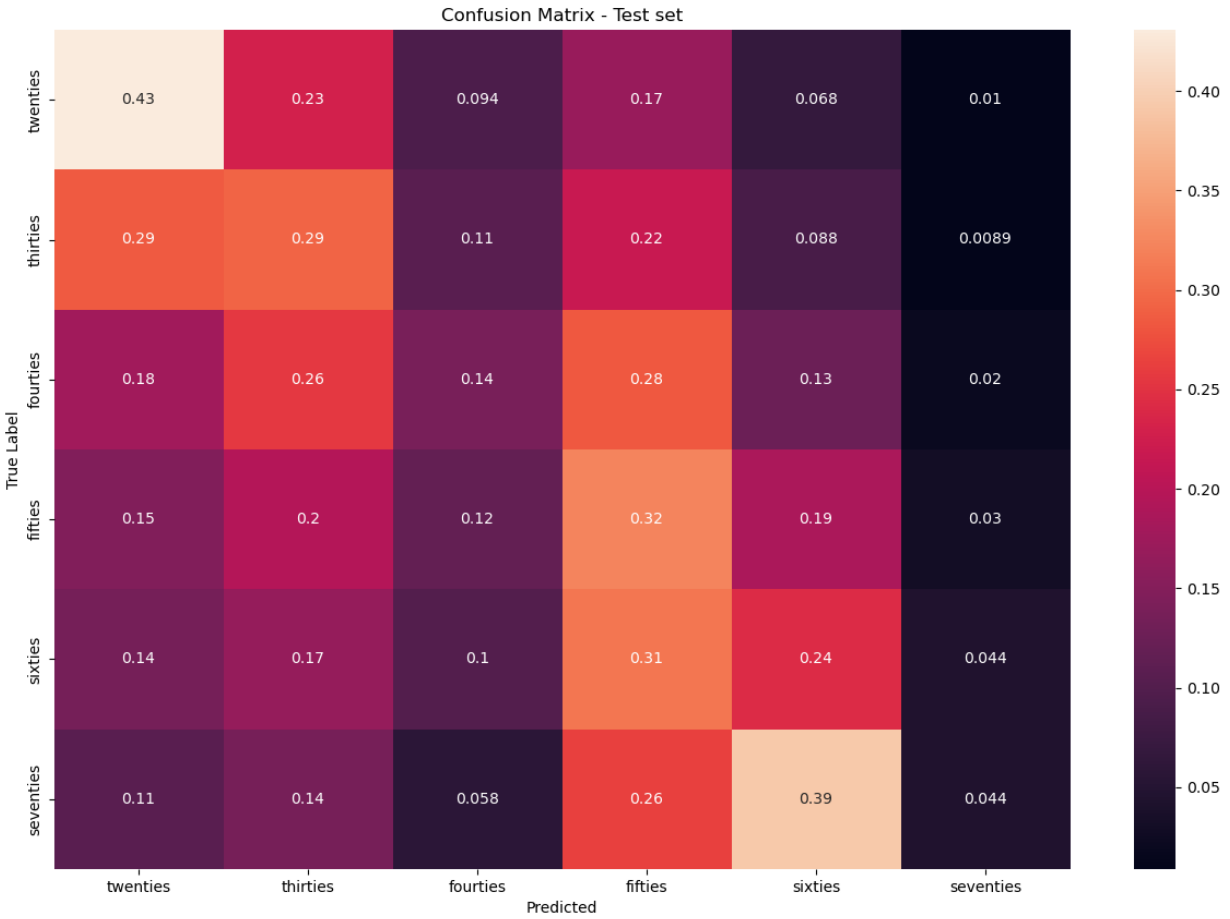

Figure 17: Confusion Matrix of AM-Softmax Combined with Ordinal Regression (Catalan)

The results of the AM-Softmax combined with Ordinal Regression are not good. The classification tasks are worst in all the classes except in the twenties and fifties. Also, the confusion matrix reflects a not proper behavior. This could be due to a lack of data that produces bad training in the combined losses. When we are training a model with more parameters, it is necessary to have enough data.

In the following table, is very clear that the first model has better performance. The three metrics show better classification and also less error distance.

Comparison between the two architectures: AM-Softmax vs Ordinal Regression (OR)			
	F-score	**Modified F-score**	**MSE (decades)**
AM-Softmax	0.250	0.639	1.224
OR	0.237	0.620	1.302

Table 3: Comparison with Catalan dataset

4.2. Results with Catalan, Spanish, and English datasets

This chapter is about the comparison of the two architectures applying the mixed languages dataset. We will see the behavior for each model with this data augmentation and decide which is the best model in a production environment.

Firstly, we are going to present the confusion matrices on the test set which are normalized over the true (rows), and then the table with the different metrics.

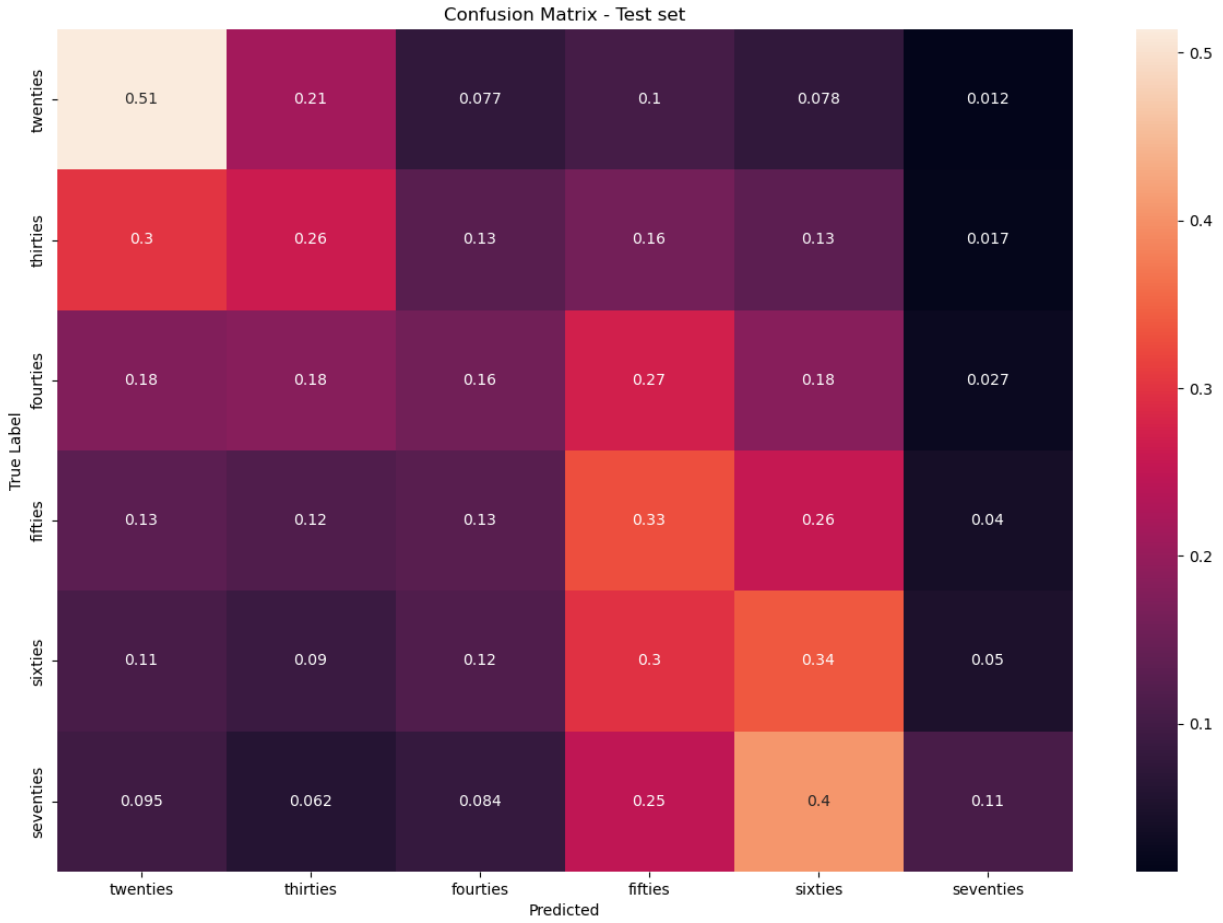

Figure 18: Confusion matrix of only AM-Softmax (ca&es&en)

As we can see in the above plot, there are significant changes in comparison with only using the Catalan dataset. Now the seventies class achieves more accuracy but less than expected, at least, the sixties class receives practically the rest. Moreover, the twenties, fifties, and sixties achieve good performance. In the two middle classes of the thirties and specially forties, the results are not quite good. For example, in the forties, practically all the classes including not neighbor classes have a huge impact on classification.

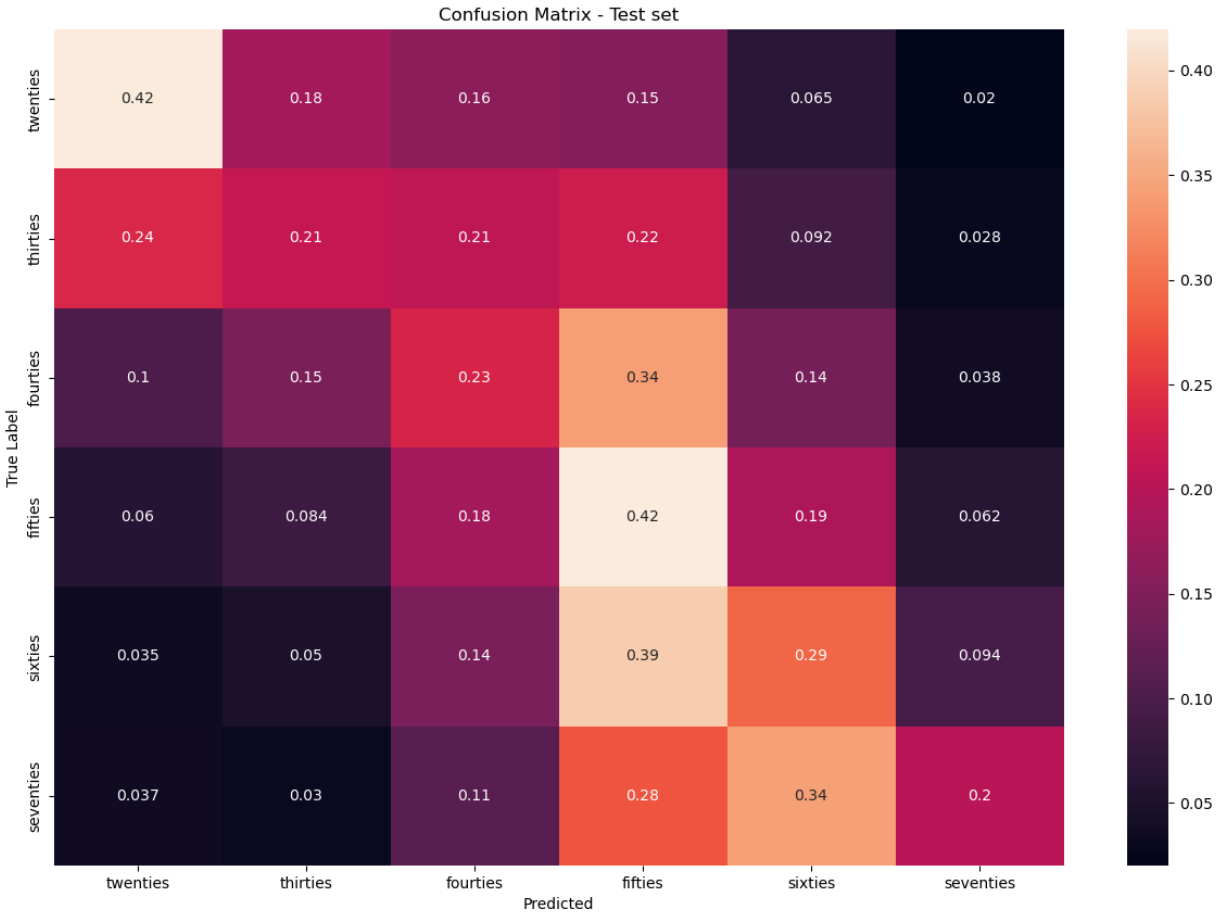

Figure 19: Confusion Matrix of AM-Softmax Combined with Ordinal Regression (ca&es&en)

With ordinal regression we have achieved the best results at the moment, the matrix is more diagonal, and in practically all the classes the classification goes into the true class or into the neighbor classes. The seventies class works better now and more than 50% of the seventies data goes into seventies and sixties classes. The classes that do not work fine in the other architecture work better in this model, the error is less and has higher accuracy. With the data augmentation, the AM-Softmax combined with Ordinal Regression has a good improvement in performance.

The three metrics tell us that the combined model works better than only using the AM-Softmax, the F-score has a great difference, and also the mean squared error is less. So, now we can say that we have a model with an error of 1.1 decades. Taking into account that the labels for training are in decades, is a good performance.

Comparison between the two architectures: AM-Softmax vs Ordinal Regression (OR)			
	F-score	**Modified F-score**	**MSE (decades)**
AM-Softmax	0.279	0.663	1.225
OR	0.296	0.682	1.141

Table 4: Comparison with Catalan, Spanish and English dataset

4.3. Feature visualization and centroids distances

This chapter consists of how the AM-Softmax output distributes the classes in a space, how they are ordered, the distance between them, the shape of the distribution, and so on.

Our purpose is to represent the features in a 3D space in order to perform it, we need to do a Principal Component Analysis transform in our data because the output of the AM-Softmax is a 6 dimension tensor so it is not possible to represent with this high dimension.

Principal Component Analysis is an unsupervised ML strategy that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components. This transformation is defined in such a way that the first principal component has the highest possible variance, and each succeeding component in turn has the highest variance possible under the constraint that it is orthogonal to the preceding components. The resulting vectors are an uncorrelated orthogonal basis set.

The representation has been done with the two architectures that have been compared in the previous section.

4.3.1. Feature visualization. Architecture 1

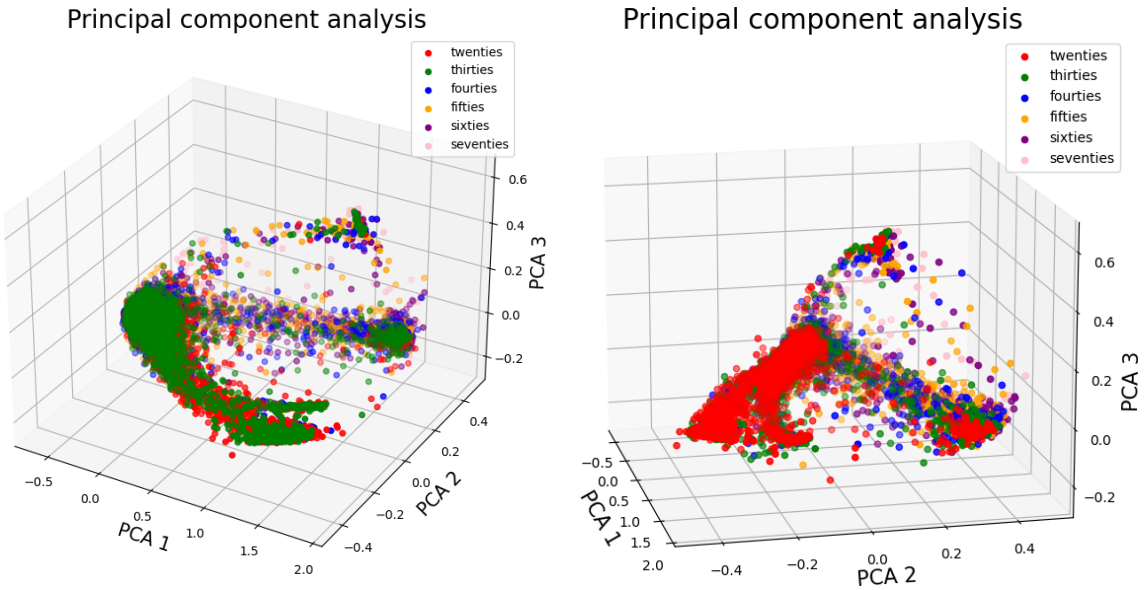

Figure 20: Principal component analysis. Architecture 1: Only using AM-Softmax

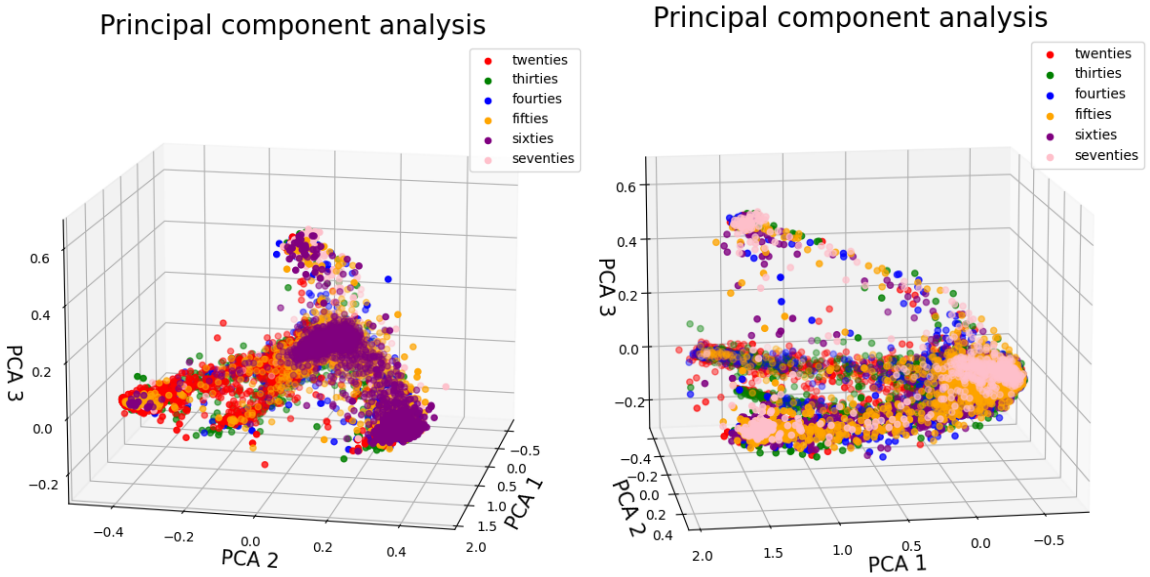

Figure 21: Principal component analysis. Architecture 1: Only using AM-Softmax

As we can see in the images, the distribution of the data is complex to visualize but still, it is possible to see how the classes are distributed in space. For instance, if we look at the

'PCA 2' axis, we can see that in the negative values, there are classes related to young people. The twenties and thirties are close between them. This phenomenon is very clear to see in the first picture.

Moreover, the classes that are of older people, are scattered in the positive values of the axis 'PCA 2', and also they are close between them. If we look at the middle ages as the forties, it is scattered into the two principal groups of the plot and is not able to be grouped by itself. This could be an explanation for the low performance in the classification task for this group of age.

4.3.2. Feature visualization. Architecture 2

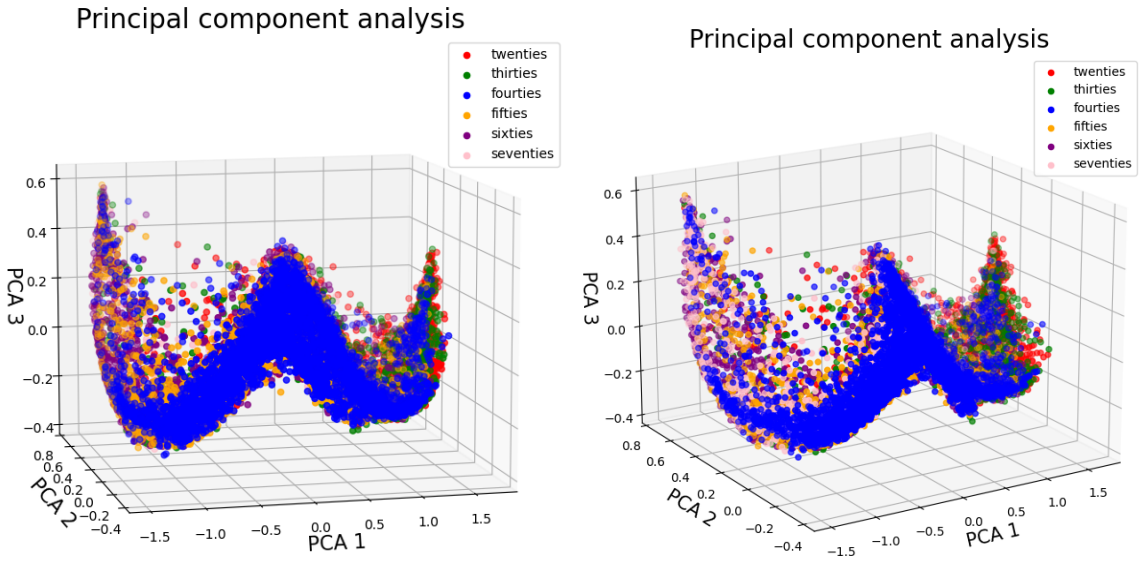

Figure 22: Principal component analysis. Architecture 2: AM-Softmax combined with Ordinal Regression

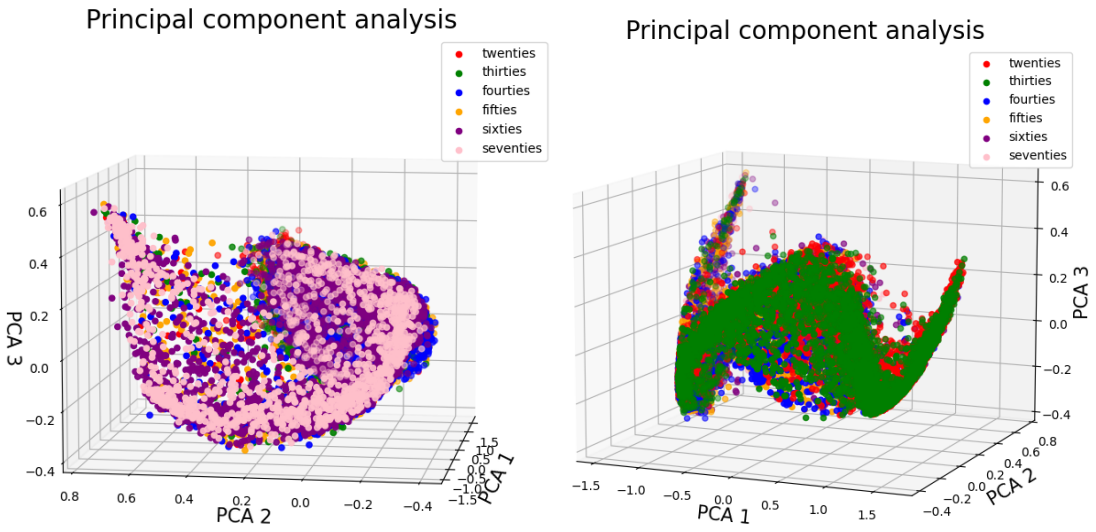

Figure 23: Principal Component analysis. Architecture 2: AM-Softmax combined with Ordinal Regression

The distribution of the data is complex as the other but also there are similarities in how the classes are distributed. In the first two images, we can see how the class of forties is between young and older ages. It separates them. Moreover, in the negative values of 'PCA1', we have the older classes closer between them meanwhile in the positive values we have the same behavior for the younger classes.

Furthermore, the distance between young and older ages is bigger than in model 1 so this model scatters the classes in a better way.

4.3.3. Centroid distances

To conclude the comparison between the two architectures, we calculate a centroid for each class in the feature space, after the output of the AM-Softmax, and then we perform the distance between them. With this information, we are able to know which model separates the classes more and this is important because it helps with the classification task.

To perform the calculation of the centroids and also the distance between them, we will use the cosine distance. Is a measure of similarity between two non-zero vectors of an inner product space that measures the cosine of the angle between them. For example, if two vectors are similar, the angle between them will be small and the cosine of the angle will be close to 1. So, since the $cos(\theta) \in [-1, 1]$ a value '-1' will indicate strongly opposite vectors, '0' independent (orthogonal) vectors, and '1' similar vectors.

The reason for using cosine distance instead of Euclidean distance is due to AM-Softmax adding a margin to help the separation between classes and this is achieved by adding a cosine similarity term in the loss function.

Cosine similarity between class centroids (Architecture 1)						
Class Centroid	**Twenties**	**Thirties**	**Forties**	**Fifties**	**Sixties**	**Seventies**
Twenties	1	0.999	0.997	0.999	0.996	0.984
Thirties	0.999	1	0.995	0.998	0.998	0.988
Forties	0.997	0.995	1	0.999	0.990	0.973
Fifties	0.999	0.998	0.999	1	0.994	0.980
Sixties	0.996	0.998	0.990	0.994	1	0.994
Seventies	0.984	0.988	0.973	0.980	0.994	1

Table 5: Cosine similarity between class centroids (Architecture 1)

Cosine similarity between class centroids (Architecture 2)						
Class Centroid	**Twenties**	**Thirties**	**Forties**	**Fifties**	**Sixties**	**Seventies**
Twenties	1	0.945	0.927	0.976	0.544	0.148
Thirties	0.945	1	0.755	0.854	0.786	0.456
Forties	0.927	0.755	1	0.985	0.203	-0.214
Fifties	0.976	0.854	0.985	1	0.358	-0.057
Sixties	0.544	0.786	0.203	0.358	1	0.907
Seventies	0.148	0.456	-0.214	-0.057	0.907	1

Table 6: Cosine similarity between class centroids (Architecture 2)

As we can see in the tables above, there is a huge difference between them. In architecture 1, the cosine similarity between the centroids is very high, so this does not help us in order to separate classes. Moreover, the decrease in similarity for far classes is very low, so this

gives the idea that only using AM-Softmax is not enough. There are classes that work better than the others but anyway, the similarity remains high.

On the other hand, for architecture 2, the similarity between centroids is lower and also decreases with the distance. For instance, if we look at the twenties, the value goes from 0.945 to 0.148. This is a very good result because demonstrates that by applying the ordinal regression we have been able to order and also distantiate the classes between them. After the results in the three metrics, feature visualization and centroids distances is crystal-clear that the combination of AM-Softmax with Ordinal Regression has better performance and also better modelization of age classification than only using AM-Softmax.

5. Budget

This project does not have the goal of creating a product or prototype, since it is purely a research-oriented one. Therefore, the cost can be calculated by adding the salaries of the participating engineers and the cost of the resources used.

To calculate the salaries of the engineers, we must differentiate between junior and senior engineers. The cost of a junior engineer can be determined by considering that this thesis comprises a total of 12 ECTS (which is equivalent to approximately 30 hours of work) and the hourly wage of a junior engineer is around 12€. Two senior engineers have participated in the development of the project so we can consider that we did a 1-hour meeting each week for fourteen weeks, with a wage of 22€/hour. The salary costs are the following:

$$C_1 = 12ECTS \cdot 30\frac{hours}{ECTS} \cdot 12\frac{€}{hour} + 2 \cdot 14weeks \cdot 1\frac{hours}{week} \cdot 22\frac{€}{hour} = 4936€$$

For the costs in resources, we have to compute the cost of the laptops and also the cost of the server. Firstly, we can consider that three laptops were used with a medium cost of 700€. For the server, we have used the UPC's Calcula platform server which could be considered as Google Cloud. The server cost is around 3€/hour and if we consider that we have done around 50 experiments with a medium duration of 10h each:

$$C_2 = 3laptops \cdot 700\frac{€}{laptots} + 50exp \cdot 10\frac{hours}{exp} \cdot 3\frac{€}{hour} = 3600€$$

To conclude, the total cost of the project is:

$$C_{total} = C_1 + C_2 = 8536€$$

6. Conclusions and future work

This study has consisted of the development and testing of different methods to perform age classification from audio records. The techniques that we have used are state-of-art in Deep Learning using mel-spectrogram. The audio records came from an open-source dataset from Mozilla called CommonVoice. Initially, only the Catalan language has been used, and then in order to have more data we added the Spanish and English datasets. It is crucial to see the data distribution, such as the number of recordings per class, establish a maximum number of recordings per user, and a speaker only can be in one of the training, validation, or testing sections.

To categorize the speech samples, we have trained and tested VGG and Resnet backbones that adhere to the same structure and concept: a front-end for extracting features from the Mel-Spectrogram, followed by an attention-pooling layer to decrease dimensionality and finally an AM-Softmax for classification. We have proved that Resnet works better in this task than VGG, it achieves better scores in the three metrics that we have used during the development of the project and also less overfitting. So, to perform the final models, we have used Resnet as the front-end.

Once, we have a baseline model to work in, the first experiment not achieved the desired results so we add more languages and also design a new model combining the AM-Softmax with ordinal regression. This model with few data does not outperform the baseline model but once we augment the data, it achieves the best results.

It has been demonstrated that this new model generalizes better the data because is capable to order better the classes and also understanding that the number of error decades affects the performance. So, after obtaining the centroids and calculating the cosine similarity we have seen a decay in similarity depending on the distance.

Furthermore, in order to compare our model with human performance, we have found a study [8] that consists of the classification of telephonic voices in age decades. The percentage of correct estimation was 27% and if adjacent classes are given as valid the precision rises to 50%. In our case, the percentage of correct estimation is 30.7%, and with adjacent classes even higher than 50%. So, compared with this study, we could say that our model is a better age classifier than humans.

Future work

The project has potential because knowing the age of a speaker could have a lot of applications such as call centers, voice assistants, and so on. For instance, it could be implemented in an AI system such as a reservation manager that refers to us depending on our age. The CommonVoice platform is a great tool that is open source to continue developing and updating the model with upcoming data.

In the future, it could be interesting to try more backbones to try to reduce overfitting and outperform the obtained results. For example, using EfficentNet could reduce the number of parameters and avoid more overfitting. To, try to select the best backbone is a good idea to visualize how the CNN interprets the input images by using, for example, Grad-Cam.

Moreover, it's necessary to continue investigating all the capabilities of AM-Softmax to separate the classes in feature space. The addition or combination of different types of losses in the Ordinal Regression could improve even more the results.

Bibliography

[1] Ardila, R., Branson, M., Davis, K., Henretty, M., Kohler, M., Meyer, J., Morais, R., Saunders, L., Tyers, F. M., & Weber, G. (2019). Common Voice: A Massively-Multilingual Speech Corpus.

[2] Miquel India, Pooyan Safari, and Javier Hernando. Double multi-head attention for speaker verification, 2020.

[3] Simonyan, K., & Zisserman, A. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition.

[4] He, K., Zhang, X., Ren, S., & Sun, J. (2015). Deep Residual Learning for Image Recognition.

[5] Hochreiter, Sepp. (1998). The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems.

[6] Wang, F., Liu, W., Liu, H., & Cheng, J. (2018). Additive Margin Softmax for Face Verification.

[7] Residual Networks (ResNet) and ResNeXtl

[8] Cerrato, Loredana & Falcone, Mauro & Paoloni, A.. (2000). Subjective age estimation of telephonic voices. Speech Communication.

[9] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention Is All You Need.