Conv-SINet: Identify Speaker using a fully-convolutional solution

sebastien.warichet
https://github.com/warichet/Conv-SINet

February-July 2020 with 24 days of partial unemployment

Abstract

The aim of my internship is to design and develop a Speaker Identification Network for Alcatel-Lucent Enterprise.
The use case could be define like that. A meeting is planed between Bart, Simon, Théo, Alban, Florian, Timothé, Michèle and Liza. Théo, Timothé, Florian, Michèle and Liza are in the same meeting room and use a octopus to connect to the remote meeting. Simon and Bart use their personal phones and are therefore easily identifiable. The problem is that when somebody speaks on the meeting room Simon and Bart haven't any information about the active speaker. So we need to be able to identify the active speaker in the meeting room.

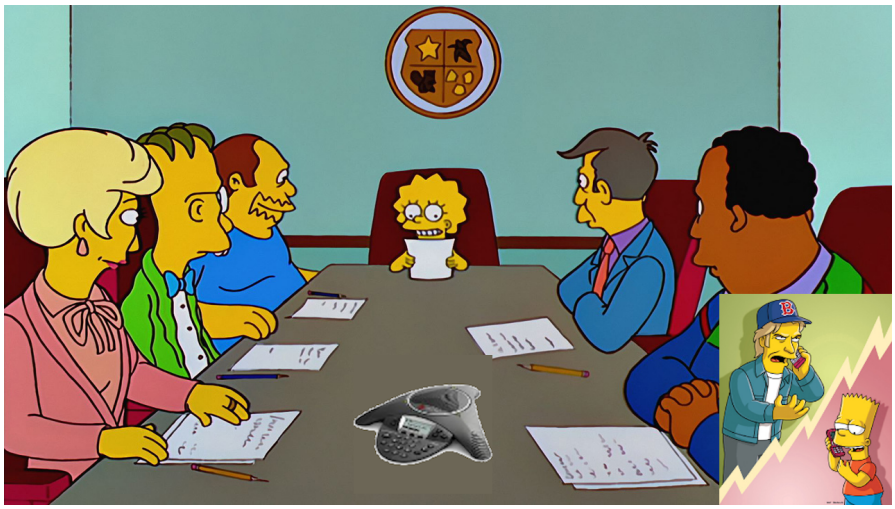

Figure 1: Meeting room

1 Background

1.1 Definition

Let's define the speakers identification problem more precisely.
Speaker recognition systems are comprised of two functional components, enrollment and recognition. The enrollment component creates reference models for each speaker from the corresponding speaker's utterance. This component is train when speakers are identified that is to say when they are connected with their own phone in our case Simon and Bart.
The Recognition component matches the sequence of feature vectors of the speaker to be identified with those reference models in the system. This component is used when speaker are under the octopus in our case Théo, Timothé, Florian, Michèle and Liza. There are two variants of speaker identification.
If the text must be the same for enrollment and verification this is called text-dependent recognition otherwise we talk about Text-independent recognition. Text-independent systems are most often used for speaker identification as they require very little if any cooperation by the speaker. This is the variant we are targeting. The text independent property could be reach with an appropriate training task (see Training objective chapter for more explanation).

1.2 Review and what news?

VoxCeleb data set is used in many project, [1] [2] [3].
Speaker identification is a very classic subject treated by multiple way, using machine learning solution or deep learning network. My choice is to combine Deep learning and machine learning techniques. Many of the previous implementations are classification problems, where the N speakers are common between the train and the test set. To simulate an encoding effective on unknown speakers test set is composed of unknown speakers.
There are designed as a mono-block with an CNN that extract the relevant features and a classifier with N class. That implementation are difficult to use in the real word because train the entire model for each company (an potentially retrain the model for each new user) is too costly.In my works the encoding and the classification are split.
The encoding is train using Siamese network, the goal is to train on as many speakers as possible to have as much generic encoding as possible. This encoding is train one time and will be used on all company. The encoding part is stored on disk and reused (transfer learning) in the next network stage.
The classification is trained by company. That is the enrolment part. Each time an speaker used is personal phone the system construct a list reference vectors. An reference vector election is done to select best vectors.
Minimum distance between vector under test and all reference vectors of all speakers is used to identify the speaker.
In previous network the sample size is fixed, most of time 3 seconds. There isn't concatenation of sample to increase the accuracy. In my network, a spacialization is used to identify a place. This place is used to concatenate voice sample. Big voice sample under test help to have a high accuracy.

2 Material

2.1 Data set

VoxCeleb is an audio-visual data-set consisting of short clips of human speech, extracted from interview videos uploaded to YouTube. For my problematic, i only use the audio part.
Due to GPU availability issues, (i works with "Google collab") i limit the train of my models to only 200 Speakers.
VoxCeleb contains speech from speakers spanning a wide range of different ethnicities, accents, professions and ages, these information are stored in metadata file.

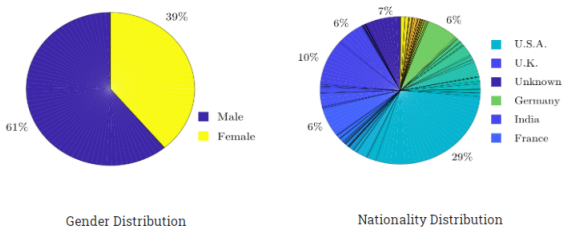

Figure 2: Data set figure from VoxCeleb site

VoxCeleb contains 100,000 utterances for 1,251 celebrities. This data-set is used for verification

and identification problematic. To be consistent with final product (we need a generic encoding efficient on speaker unknown to the system), train set and test set are made up of different group of speakers aka although our problematic is identification, we use the verification split.

Verification	dev	test
# of speakers	1,211	40
# of utterances	148,642	4,874
Identification	dev	test
# of speakers	1,251	1,251
# of utterances	145,265	8,251

Table 1: Split

VoxCeleb1 is fairly gender balanced (55% male). The speakers span a wide range of different ethnicities, accents, professions and ages. The auditory environments are multiple and real-world. If the test-data set is not so well balanced, it is a very large data-set.
I thinks there is a bias in VoxCeleb1, unless the environments are multiple, at speakers level the environment could be very uniform. In this case the encoder may be tempted to use this noise. That could explain a part of very good result if identification split is used (See id10024 speaker in dev split).
Another problem is that the speakers are sorted by first name. When i have split the training data-set to do a cross validation, i ended up with a split consisting only of "David". In other words, a split with very similar speakers (male, Anglo-Saxon ...), unsurprisingly this lowered my accuracy.
In the near future, there are plans to check my model on another data-set. POLYCOST and SWB data-set are good candidate due to their telephony condition.

2.2 Google Colab pro

At Alcatel there is no GPU available and no budget to buy it. So the only way to access GPU is Google colab pay on my personal money. The pro version give access to GPU T4 and P100. That's fine, but there are limitations especially on idle times disconnection. On this solution it is not possible to train the encoder on more than 200 speakers, unless the data-set have more than 1000 speakers.

3 Methods

3.1 General architecture

In this section we will present the architecture chosen to solve this problem.
A big part of architecture is base on the fact that CNN extracts pattern. The best way to identify speaker is the tracking of pattern between the referenced voice samples (extracted during enrollment) and the voice sample under octopus. To increase the accuracy of our system we must increase the probability of matching pattern. To do that we used multiple reference vectors and we concatenate samples under test.
This architecture could be broken down into three main parts, the encoding, the enrollment and the recognition part.

3.2 The encoder

The encoder part extracts the feature vectors from speech input using CNN block.
The encoding should be as generic as possible. In short, the encoding must be effective on unknown speakers, this explains why the verification data set is used. That also could explain the relatively modest results of the encoding phase.
The Siamese training strategy is the most effective for this type of problem. This encoder is trained one time, all companies will use this same encoder.
We have tried two different methods for this part, one based on the frequency spectrum of the signal and the other based on a fully time domain.
The training strategy
For learning by triplet loss, an anchor vector is compared against a positive vector and a negative vector. The goal is to minimize the distance between anchor and positive vector and maximize the distance between anchor and negative.
A current loss function for Siamese network is:
Let's $d(a, n)$ the distance between anchor and negative.
$\mathcal{L}(a, p, n) = Relu(d(a, p) - d(a, n) + margin)$
For more details about the encoding A

3.2.1 Transformation of row input

To improve accuracy a transformation is performed on input, the DC component of the signal is removed. Although the DC component is very small, i noticed a real impact on my results. According to wikipedia, in audio recording, a DC off-

set is an undesirable characteristic of a recording sound. It occurs in the capturing of sound, before it reaches the recorder, and is normally caused by defective or low-quality equipment. This bias is easy detected by the network (especially in fully temporal domain) and distorts my results.
The amplitude of the signal is reduced between -1 and 1.

3.2.2 Compare vectors

Once a vector is generated, this vector must be compare to identify the speaker. To do this job, we need a data base where are stored one (or more) vectors by speakers. The speaker which have the shortest distance between his stored vector and is chosen.
The identification of speaker is performed by minimizing the distance between the vector under identification $S \in \mathbb{R}^{\mathbb{N}}$ and the stored vectors $S_i \in \mathbb{R}^N$.

Euclidean distance
Let's use euclidean distance to identify the speaker.

$$i = Argmin_i(\sqrt{\sum_{n=0}^{N}(S_{in} - S_n)}) \tag{1}$$

To reduce the cpu consumption sqrt calculus should be ignored.

Cosines similarity
For cosine a value of -1 define opposite vectors, 0 independent vector and 1 similar vector.

$$i = Argmin_i \left(\left| \frac{S.S_i}{\|S\|.\|S_i\|} \right| - 1 \right) \tag{2}$$

Euclidean distance is selected due to best accuracy 0.863 vs 0.792.

3.2.3 The encoder in frequency domain

This encoder is inspired by VoxCeleb and VGG architecture. Like VoxCeleb implementation, a Short-time Fourier Transform (STFT) is done on voice sample. That transform 1D voice sample to 2D sample representation in frequency domain. An important characteristic that the image must have for effective CNN analysis is that nearby pixels must share information. It is the case for image generated by STFT.
To train the network, i used a length of 250 such as just 1 second of sample voice. I have try different length for sample, from 3 seconds to 1 second, the accuracy values are comparable. The advantage of short length is to increase the genericity of the encoder and prevent the over-fitting. With short sample, the probability to detect the same simple pattern (for example a same word in the anchor and positive sample) decrease. See result of identification chapter 6.3.

sample length	train accuracy	test accuracy
1	0.922	0.866
2	0.941	0.874
3	0.958	0.886

Table 2: Accuracy vs sample length in frequency domain

Due to hamming windows size equals to 25 ms and a hop length equals to 10ms, an overlap is generated in this transformation.

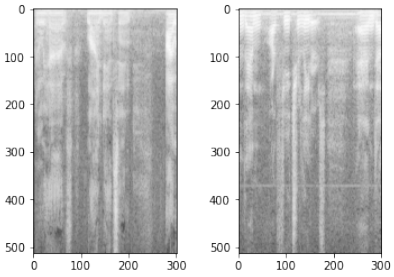

Figure 3: STFT images

Four convolution block are applied on these created images. These block according to VGG architecture are a stack of one convolution, an activation (Relu), a normalization and a polling layers. For more detail see appendix B.

3.2.4 The fully time domain encoder

The second encoder has the same structure as the first except the CNN which becomes 1D.
The time domain encoder have more works, it must generate a dictionary equivalent to the frequency decomposition. This dictionary should be very effective because it follows directly from our problematic. But it requires a lot of data, a lot of

processors, and a deeper and less stable network.

sample length	train accuracy	test accuracy
1	0.866	0.746
2	0.884	0.778
3	0.901	0.796

Table 3: Accuracy vs sample length in time domain

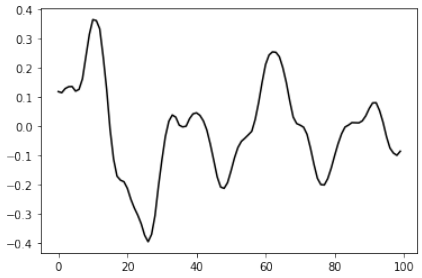

Figure 4: Signal in time domain

3.2.5 Encoder conclusion

The encoder that use frequency decomposition of signal have a better accuracy. It is due to a best generalization of the encoding. A cross validation is done to calculate mean and standard deviation on accuracy. The cross validation use the same encoder on 10 bags of 20 unknown speakers. For more explanations about cross validation see C.

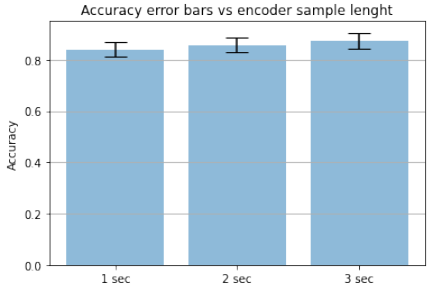

Figure 5: Error bar for frequency encoder

The encoder that use the signal in time domain have worst accuracy and more overfitting. The best accuracy in temporal is for sample of 2 seconds. This accuracy is insufficient for Alcatel's needs but is not without meaning because it is far from being random. We notice a bigger standard deviation between 0.039 and 0.048 instead of 0.027 and 0.030 for frequency encoding. That is coherent with less generalization and more over-fitting.

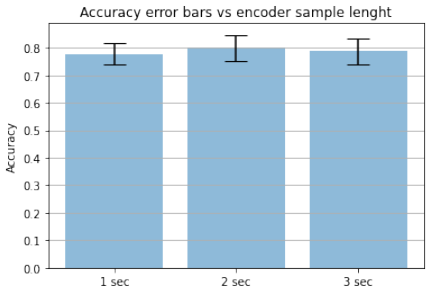

Figure 6: Error bar for frequency encoder

The election and identification use vectors generated by the encoder. To do that, encoding network is saved into a file and reuse by other part of network following the principle of transfers learning.

3.3 The enrollment part

The enrollment select pertinent vectors generated when speaker is identified (user under personal phone). In our use case Simon and Bart are the identified speakers and they are the speakers involved in this election. These reference vectors will be used for the recognition part to identify the speaker.

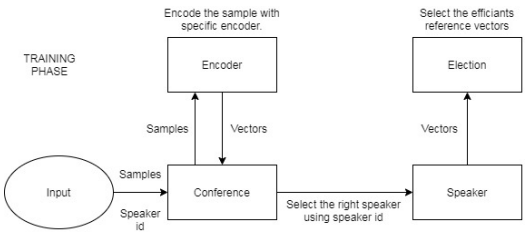

Figure 7: Training

3.3.1 The election

At start twenty reference vectors are used for each speaker. The used of multiple vectors increase the probability of pattern matching.
Each reference vector have a score. For a speaker, the network calculate distance between reference vectors and vector under analysis. A ratio is calculate with this calculus $ratio = min(10, \frac{worstdistance}{bestdistance})$. The worst vector win ratio points. The best vector loss ratio points. If a reference vector reaches a positive threshold, it is removed and replaced by the vector under analysis. By definition the vector under analysis is far from the removed vector, that make him the ideal candidate.

3.3.2 The exploration vs the stabilization

At the start of the election, we have to integrate a lot of new vectors into the reference vectors pool to explore all the possible candidate.
After a time, we needs more stabilization because new vectors have a low probability of being optimum. Without stabilization, after a peak of accuracy we notice a loss of performance.
The parameter use to manage this balance is is the minimum score below which a vector is eliminated. This threshold is between -100 and -1000. If the minimum is closed to -100 the vector still in danger (exploration) and could be removed is it won to much point. If the minimum is closed to -1000 the probability to remove this vector is very low (stability).
Each time the label corresponding to the mean vector of the speaker doesn't matches with the gold label, we call the "explore()" method that will grow "down" the threshold. On the contrary if mean label correspond, the threshold will decrease, that correspond to a stabilization of the election for the speaker.
To force the new vectors to be different from the reference vectors pool, we only remove a vector if the current ration is less than 2. Indeed if the ration is too small that indicate the new vectors is very similar to the reference vectors. That avoid having very similar vectors in reference pool.
In the same way, we only remove the vector if the distance is large enough.

3.3.3 Enrollment conclusion

In term of enrollment, the baseline correspond to a pool of size equals to one without election. This configuration done an accuracy of 0.5037. With 20 vectors without election, the accuracy rise to 0.7596. At the end, the election brings the accuracy to 0.8603.

pool size	start acc	max acc
1	0.5037	0.5037
2	0.5454	0.5543
5	0.6785	0.6968
10	0.7179	0.7896
20	0.7596	0.8603

Table 4: Pool size vs accuracy

Using multiple vectors in reference pool the probability of matching pattern rise. If we select the best vectors (efficient and different from each other) in the pool, the accuracy begins to be acceptable.

3.4 The recognition part

To determine the active speaker, the recognition part compares vectors generated in conference room (user under octopus) with reference vectors pool previously generated (user under personal phone).The short distance win.

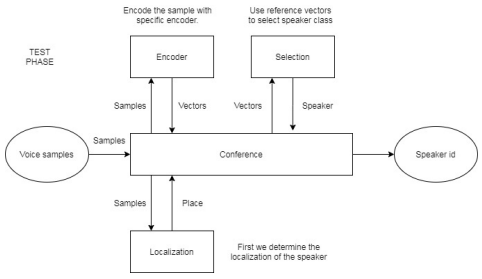

Figure 8: Identification

3.4.1 The best vector vs the topk vectors vs the mean vector

For each simulation we try three strategy. The first strategy is to keep the best distance, the second is to keep the k best distance and the last is the get the mean distance.

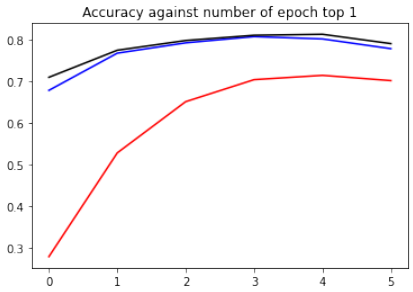

Figure 9: 20 speakers frequency accuracy vs strategy: red=mean blue=top 4 black=best

The black and blue curves are very closed, the red curve is 0.1 bellow. We can easily explain this by the fact that we are looking for patterns. One part of the 20 distances we have have matching pattern and the other part is without matching pattern. If we take the mean of all distances the worst vectors dilute the information of the best vectors. The best result for topk is for k=4, That indicate at list 4 ref vectors have corresponding patterns.

3.4.2 The impact of encoder sample size

Let's classify 20 speakers with 3 different frequency encoder. The first encoders (encoder trained with sample of 1 second) give best results. The 2 other (2 and 3 second sample) have the worst result. This verifies the hypothesis that an encoding with short sample give a more generic encoding. The less the probability of matching a pattern during training, the more the network learn generic feature.

sample length	min acc	topk acc	mean acc
1	0.871	0.853	0.784
2	0.808	0.813	0.719
3	0.808	0.825	0.754

Table 5: Accuracy vs encoder sample length in frequency domain

In time domain the accuracy is far from being random (0.606 instead of 0.05) but not sufficient for our purpose.

encoder length	min acc	topk acc	mean acc
1	0.595	0.597	0.525
2	0.585	0.606	0.538
3	0.548	0.587	0.526

Table 6: Accuracy vs encoder sample length in time domain

3.4.3 The localization

Why we need localization?
The localization part uses the three microphones to determine the place of the speaker.
Once the sample are localized, we can concatenate the samples for each place and increase the performance of network (on this first approach we consider that speakers are static). The used of concatenate vectors under identification increase the probability of pattern matching.
Like identification, the localization could be decomposed into three main parts. The specific encoder, a selection of point of interest and a concatenation of multiple results in a time window.A mechanism that calculate the delay between microphones should be added.
"pyroomacoustics" library is used to simulate a two mics room environment. The "voxCeleb" data-set is used as input of this simulation.
The position of the speakers is chosen so that the two audio streams are without delay. In other words, the speakers are equidistant from the microphones. For each mini batch treatment, a new room is simulated. Each simulation have different place for microphones and speakers, that is important to simulate different case of echos.

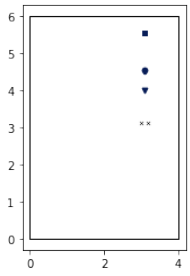

Figure 10: Room simulation

Due to lack of time this part is simulate. Only the encoder is implemented.
The localization encoder is a Siamese network that generate vectors. The loss function verify that the distance between vectors corresponding to same sample is smaller than distance between two close samples (in an interval between -10 and + 10 centered on the samples). There is pooling in the CNN, that keep time line accurate.

vector length	accuracy
16	0.9680
32	0.9775
64	0.9774
128	0.9827

Table 7: Accuracy vs vector length

The next point is selecting point of interest. This network should make it possible to select the most efficient vectors. The data-set is labelled during the encoding, each time the Siamese network return a distance under a threshold the vectors are labeled inefficient. Otherwise the vectors are labeled efficient. So we have a DNN classifier which has vectors as input and 2 classes as output (efficient vector or not). Last point is the calculus of delay. Once efficient vectors are selected, we need to use those vectors to determine the delay between mics. The networks concatenate the results obtained by selected vectors to increase the delay accuracy. In the end, we consider that different delays are generated by different places.

3.4.4 Recognition conclusion

In a meeting a person could talks some minutes at the same place, this fact makes the following results relevant to our problem. The accuracy come from 0.8603 (best accuracy for one sample of 3 seconds) to 0.988 (best accuracy after 3 samples of 3 seconds), thanks to samples result concatenation.

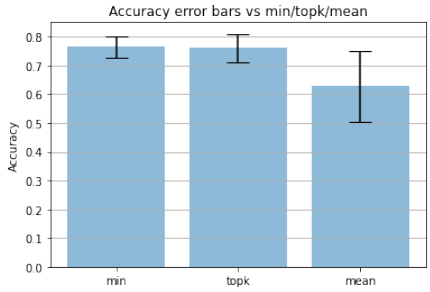

Figure 11: Error bar for one sample under analysis

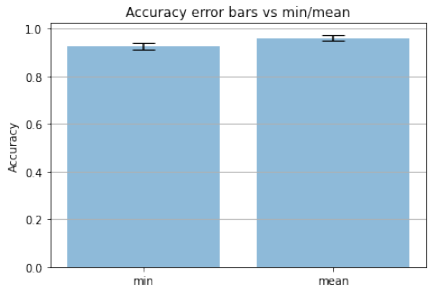

Figure 12: Error bar for three sample under analysis

Once we have more sample under analysis, it possible to reduce the reference pool size. So that allows us to reduce the cpu consumption.

# vectors in pool	1 sample accuracy	3 samples accuracy
5	0.571	0.891
10	0.664	0.915
15	0.683	0.950
20	0.766	0.988

Table 8: Accuracy vs # samples under analysis vs # vectors in reference pool

4 Results

For a meeting of twenty speakers, we can notice an accuracy between 0.90 and 0.95. The top accuracy become after only 2 epochs and decreases until 0.90.
For the encoder, however the full time domain learn a dictionary directly from data-set, frequency decomposition of the signal gives the best results with approximately 10 percent less for full time domain. In an ideal world, it should give best results but it requires a more complex network to train. Indeed, the difficulty of this problem is to have the most generic coding possible (By generic encoding I mean an encoding where the same speaker generates close vectors regardless of what is said). It is so easy for the network to match the same sound emitted by the same person instead of having more pitch and timbre based encoding. This is why frequency coding is easier to train and a short vocal sample gives the best result at the end of the process. It is validated by the results, with short sample (one second) the encoding accuracy is worst but this encoding give best results when it is used for identification. See Table 2 5. Another clue that indicates that networks are getting easier is the speed of learning. With small sample sizes the learning is slower which may indicate a more complex learning.

sample size	best test accuracy	best epoch	0.86 epoch
3	0.886	25	5
2	0.877	21	8
1	0.863	32	32

Table 9: Speed to reach the maximum accuracy or 0.86 accuracy

In confusion matrix we have only one speaker with a very small score. It could be due to election, in election each speaker work alone to have the best reference vector. Let's study the case of speaker 285 vs 288. At the start the speaker 288 have a full accuracy but on top that it generates 67 percent false positive on speaker 285. At the end we have 75 percent of false positive. The election should solve this problem with a penalty if the vector is too efficient on other speaker. Thanks to confusion matrix.
See appendix C for confusion matrix.
Despite everything, as the cross validation has shown us, the results must be put into perspective by the fact that a meeting can include speakers with very similar characteristics.For example a meeting with a majority of (wo)men from the same country and the same social environment.

5 Conclusion

The first lesson is that mixing deep learning (encoding) and machine learning (election) could give very efficient product.
Another point is that the deep network is lazy and sometimes it might help to make it more difficult for them. And this in order to achieve a more complex (deep) learning. See sample size impact in Results.
An network architecture composed of a specific encoder, a selection of point of interest and a concatenation of results could be efficient on different problems. In this stage it is the general architecture and it is also the architecture for localization problem.
To put an end to more down-to-earth problems.
In theory (if localization works fine), an accuracy of 0.95 could be archived. This is a result inline with Alcatel requirement.
But due to lack of time (partial unemployment), there is still some work to do.
We need to resolve the localization.

It was also noted that the election system can be improved (see results).
In this first approach, the inputs are not meant to be a mix of multiple speakers. A Speech Separation block could be added.
If there is no problem to develop and train the model using python, an integration of recognition in Alcatel products requires a real language like C++ or JAVA (that is possible to convert the network in pytorch script, this script is serialized and load on C++ module).

Once this problem is solved all voice samples can be tagged by user. This tag will be used on other products like "Automatic research of information in conference server".

References

[1] A. Nagrani, J. S. Chung, A. Zisserman VoxCeleb: a large-scale speaker identification dataset INTERSPEECH, 2017

[2] Shaojin Ding1, Tianlong Chen1, Xinyu Gong1, Weiwei Zha2, Zhangyang Wang1 AutoSpeech: Neural Architecture Search for Speaker Recognition May 2020

[3] Arsha Nagrani1, Joon Son Chung1, Weidi Xie1, Andrew Zissermana Voxceleb: Large-scale speaker verification in the wild May 2019

[4] Yi Luo, Nima Mesgarani Conv-TasNet: Surpassing Ideal Time-FrequencyMagnitude Masking for Speech Separation 2019

[5] Arsha Nagrani†, Joon Son Chung†, Andrew Zisserman Visual Geometry Group, Department of Engineering Science, University of Oxford, UK arsha,joon,az@robots.ox.ac.uk

[6] T. Muto and M. Sugiyama, "Model based voice decomposition method with time constraint," 2001 IEEE Fourth Workshop on Multimedia Signal Processing (Cat. No.01TH8564), Cannes, 2001, pp. 21-26, doi: 10.1109/MMSP.2001.962705.

Appendices

A More detail on encoding architecture

In this appendix we will present the frequency encoding and the full time domain encoding.

A.1 Architecture

The first level of block is used to generate features. From SFTP image (or a voice sample), it produce 256 features.
The next level of block is in charge of combining the 256 features between them and add complexity to the ending features. There is no maximum pooling in these blocks, the sorting of the features which were carried out in the preceding blocks.
After blocks an average pooling gives reduce the time line to 1. So we have for each sample, an ending result in the form of a vector of size 256.
The last convolution with a kernel size equal to one, is responsible for giving an adaptive weight to each feature.

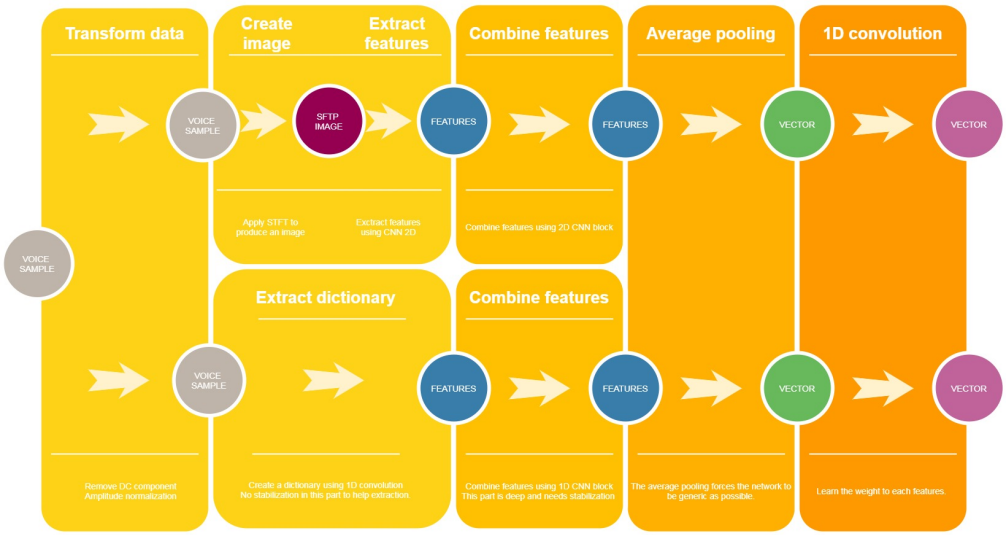

Figure 13: Convolution block

A.2 Convolution block

The convolution block is the basic brick of this neural network. This block is a variation of Conv-TasNet research [4].
There are two version of this block, a 2D version (that use 2D CNN) used for frequency purpose and a 1D version use in time domain.

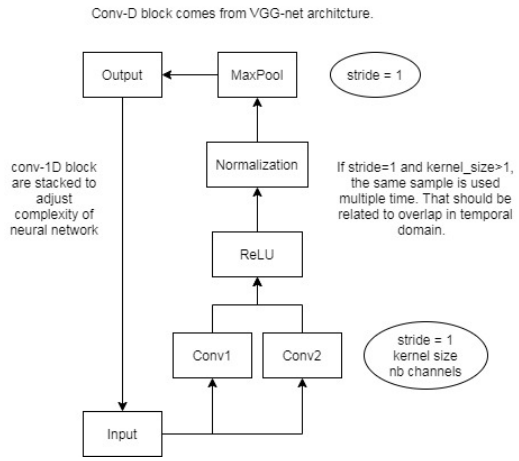

Figure 14: Convolution block

An inception mode could be activated in the block. Our inception consists of two convolution in parallel. The first convolution have a kernel size equal to one an generate one quarter of output features. That play a role of stabilization. If a problem occur in convolution at least one quarter of output features are very closed to the input feature. The second convolution have a bigger kernel size and generate the other tree quarter of output features. The role of this convolution is to generate new features. In a general way i try to limit the padding in convolution, but in inception i was forced to add little padding.
The max pooling is used to select in time the best output features and limits processor consumption.

B Dynamic localization of speakers

On the first implementation, the speakers are considered static, but it could be different in the real world. To deal with this problem, we have one solution.
The first solution is to use the variance of the voice fingerprint. This solution could be compatible with a mono-microphone environment.
In this case a place could be defined as part of voice samples with a small voice fingerprint variance. According to the temporal continuity constraint, for a single speaker, the given spectral sequence changes are smoother [6]. Let's $s(t-1) \in Class(Liza)$ a sample of voice at time t-1 classified in Liza class.
The best probability for $s(t)$ is the class Liza corresponding to the case where Liza have not finished to talk at t-1 time.
After that the best probability should be "Florian" the last speaker before Liza corresponding to a dialog between Liza and Florian.
If "Michèle" never talk during meeting, we can consider that the probability for $s(t)$ is in Michèle class is very low.

In term implementation that should look like that. For a place L if we have a huge variation in the fingerprint at time t, we can consider that the speaker is no more Liza. It's time to check if place F corresponding to Florian have a small variance. This process continues until place M corresponding to Michèle.
Additional information could reduce the possible configurations. For example, we know that there is one location per user and the speaker can only be in one place. This information could be useful to

follow the movements of the speaker and dynamically update the location of the speaker.

C Error bar

We have lot of data but few GPU time. Starting from this state of affairs we have modified the principle of cross validation.
We have take ten pools of twenty speakers unused in train set. After that we have used the trained encoder. That is different to classic cross validation where the model is retrained for each pool. I get the error on the trained model instead of the general model. This is consistent with our use of the encoder which is only trained once.
Despite everything we encountered a problem with the composition of the test set. The speakers are sorted by first name, the first split consisting only of "David". In other words, a split with very similar speakers (male, Anglo-Saxon ...), unsurprisingly this lowered my accuracy. But This is also consistent with our problematic. It is very rare to have meetings with very different people who speak different languages.

D Confusion matrix

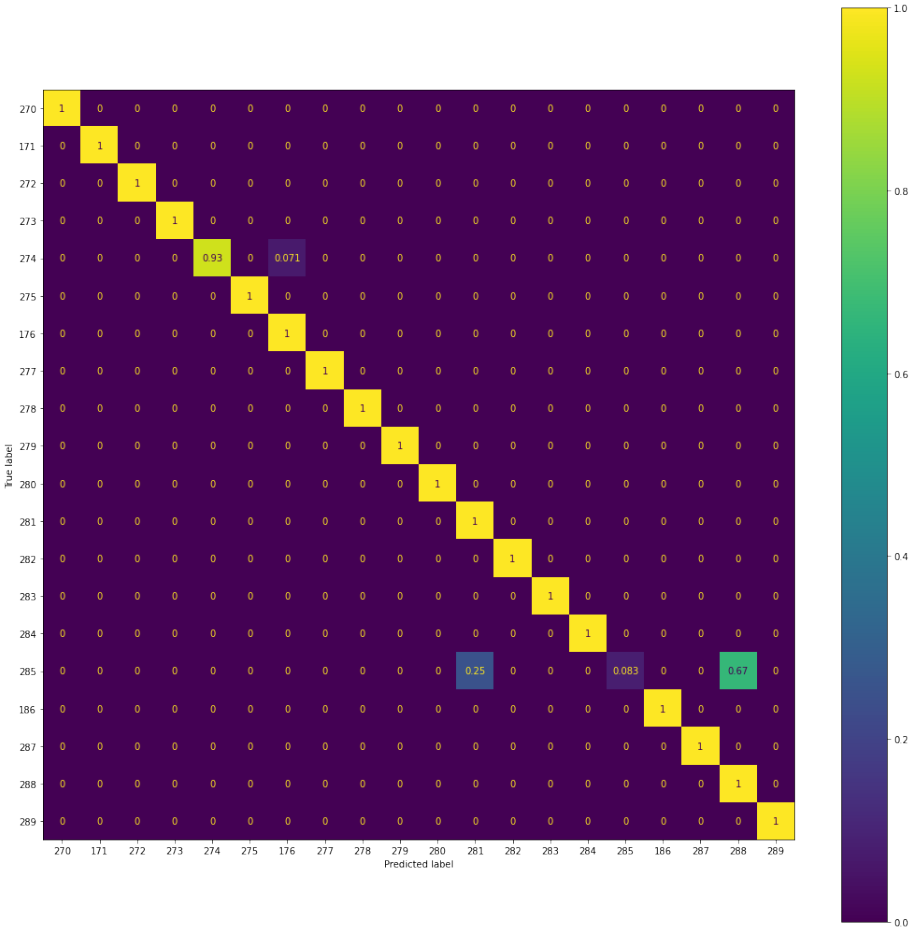

Figure 15: Top confusion matrix

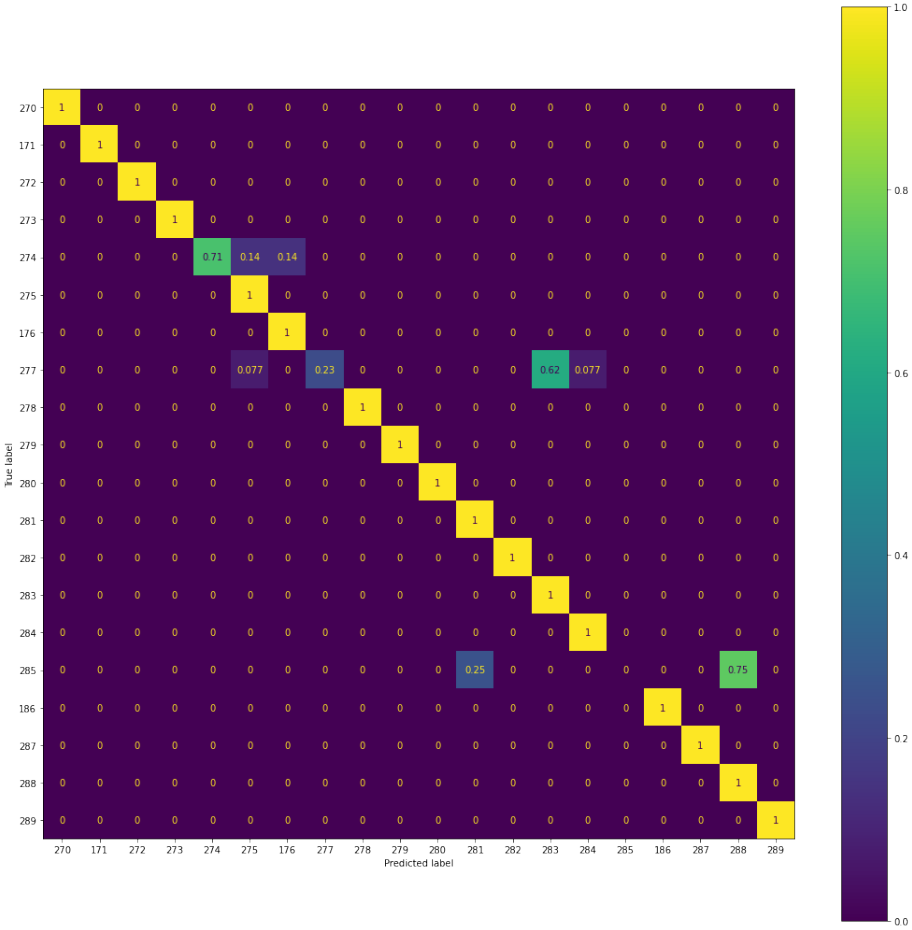

Figure 16: Final confusion matrix