LABEL REFINING: A SEMI-SUPERVISED METHOD TO EXTRACT VOICE CHARACTERISTICS WITHOUT GROUND TRUTH

Anonymous authors
Paper under double-blind review

ABSTRACT

A characteristic is a distinctive trait shared by a group of observations which may be used to identify them. In the context of voice casting for audiovisual productions, characteristic extraction has an important role since it can help explaining the decisions of a voice recommendation system, or give modalities to the user with the aim to express a voice search request. Unfortunately, the lack of standard taxonomy to describe comedian voices prevents the implementation of an annotation protocol. To address this problem, we propose a new semi-supervised learning method entitled Label Refining that consists in extracting refined labels (e.g. vocal characteristics) from known initial labels (e.g. character played in a recording). Our proposed method first suggests using a representation extractor based on the initial labels, then computing refined labels using a clustering algorithm to finally train a refined representation extractor. The method is validated by applying Label Refining on recordings from the MassEffect 3 video game. Experiments show that, using a subsidiary corpus, it is possible to bring out interesting voice characteristics without any a priori knowledge.

1 INTRODUCTION

To understand the world aroung us, the AI must fundamentally "The goal of representation learning is often to make learning algorithms less dependent on feature engineering so that the novel applications could be constructed faster" Bengio et al. (2014).

Represent In order to propose a way to, this paper introduce a new problematic consisting. So, we propose a formal generic expression of this problematic and a generic method to solve it.

We generaly associate data representation to features extraction. here, we talk about a specific aspect of features we call characteristics. While these two terms are quite similar, we associate a formal difference. Where features are dimensions with a real value associated to each observation, characteristics are hard labels where values are in 0, 1.

We consider Label Refining as a semi-supervised learning approach because it uses labels in order to learn representation but these labels are not the final one we are looking for.

Developping label refining could be useful like for Heimerl et al. (2019).

Annotating enough data can be a lengthy and cumbersome task. In this experiment, classifier is trained on manually labelled examples to optimise a learning function. Once trained, the classifier is used to automatically predict labels on unseen data.

Increase the transparency of the decision process. Contrairement à ce travail, nous ne connaissons pas les labels qui vont aider à décrire les données. Nova Heimerl et al. (2019) propose d'annoter automatiquement les données et de les faire vérifier par les experts. Cependant, dans ce cas, la définition des labels est connue par avance.

A good representation is often one that captures the posterior distribution of the underlying explanatory factors for the obsreved input. We base on this sentence in order to say that our new representation should help to classify on a downstream task.

What are appropriate objectives for learning good representation?

Two mains branches of representation learning. Supervised learning. Unsupervised learning.

Quelque soit la Representation Learning deals with representing with features extraction where each feature is a dimension, often represented with a number belonging to the real mathematical domain.

In this article, we differenciate the notion of features from the notion of characteristics. The definition of these features is directely related to the definition of typicality since it serves to discriminate observations or group of observations. For example, we want to differenciate recordings belonging to the same speaker in speaker identificartion and we need to represent these recordings through a vector representation that helps to do it.

In order to test it, we propose to adapt the method to context of dubbing voices.

As petroleum refining refers to all treatments and transformations aimed at extracting from petroleum the maximum of products with high commercial value, we propose to name our processus label refining.

2 LABEL REFINING

As the only knowledge available is a label, . we propose a formal way to see this. We call it label refining. In this section, we propose a generic formulation and method to refine labels in sublabels. Based on neural networks and clustering algorithms.

2.1 THE DEFINITION OF REFINED LABELS

In this work, we assume that labels are decomposable into different lower-level sublabels. In our case, from the initial label of the character associated with a recording, we wish to decompose the character into vocal characteristics such that the emotion played (e.g. agressive, happy), a voice color (e.g. white, red) or a timbre (e.g. husky, enlightened).

The equation 1 formalize the definition of refined label. We note R the set of refined labels and $a_i^{(l)} \in \{r_1, r_2 ... r_{|R|}\}$ the ith refined label associated to the initial label l. Let us also specify that the number n of refined labels is different depending on l. Thus, from an initial label l we wish to find the refined labels $a_i^{(l)}$.

$$l \begin{cases} a_1^{(l)} \\ a_2^{(l)} \\ ... \\ a_n^{(l)} \end{cases} \tag{1}$$

Let us note the functions $f : e \mapsto l$ and $g : e \mapsto r$ where o is an observation (e.g. vocal recording). The equation 2 gives fictives examples. It shows the expected images of the functions f and g according to two initial labels l_1 and l_2. In our model of the problem, we assume that a refined label r may appear in observations associated to different initial labels. Here, this is the case with r_1 which appears both in an observation associated to l_1 and an observation associated to l_2.

$$l_1 \begin{cases} f : o_1^{(l_1)} \mapsto l_1 \\ f : o_2^{(l_1)} \mapsto l_1 \end{cases} , l_1 \begin{cases} g : o_1^{(l_1)} \mapsto r_1 \\ g : o_2^{(l_1)} \mapsto r_2 \end{cases} , l_2 \begin{cases} f : o_1^{(l_2)} \mapsto l_2 \\ f : o_2^{(l_2)} \mapsto l_2 \end{cases} , l_2 \begin{cases} g : o_1^{(l_2)} \mapsto r_1 \\ g : o_2^{(l_2)} \mapsto r_3 \end{cases} \tag{2}$$

2.2 A FIRST METHOD THAT WE PROPOSE TO REFINE LABELS

Figure 1 presents the refined label extraction method that we propose in three steps. 1) consists in training a system to extract a representation (i.e. vector, hashRoussev (2011)) of the observation $o \in O$ using the initial labels $l \in L$. We illustrate this step with a neural network embedding. 2) consists in using a clustering algorithm in order to associate new labels $r \in R$ to the observations O. 3) consists finally to train an new system to extract refined labels by using labels r.

In order to evaluate the relevance of labels r, we use two methods. Applied to the step 2, the first method is to apply different clustering quality measures such as v-measure or the purity. Applied to the step 1 and step 3, the second method consists in extracting a representation of the observations O and evaluating it on a downstream task. This task can be diverse: speaker recognition, speech recognition or, in our case, character voice similarity. Thanks to the second method, it is possible to evaluate the obtained representation from the step 1 and to compare obtained results with the step 3. Thus, we can highlight the influence of the refining process on the information encoded by the systems.

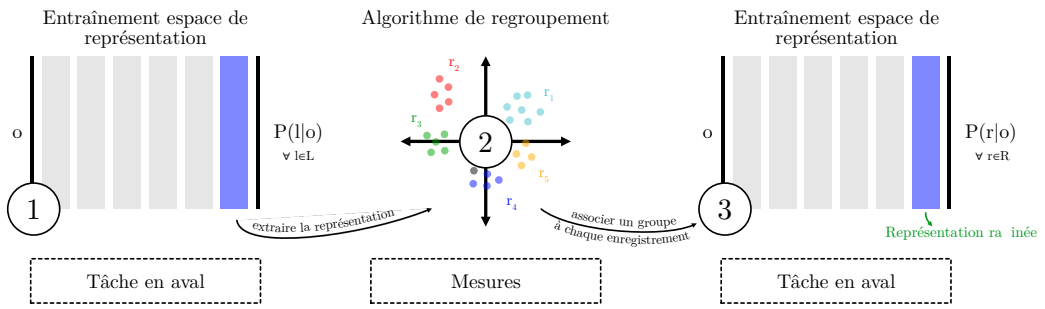

Figure 1: Refining process of labels $l \in L$.

3 Experimental protocol

We propose to adapt the label refining we introduced for the character representation. This consist to use p-vector in order to extract voice characteristics.

3.1 Main (MassEffect) and second (Skyrim) Corpora

We use MassEffect data as the main corpora. We also describe a second dataset from the video game Skyrim that we use as a validation set.

3.1.1 MassEffect, the set used to train and evaluation p-vectors

The main corpus is composed of voice recordings coming from the *Mass-Effect 3* role-playing game. Contrary to movies, voices of video games present some particularities explained in Gresse et al. (2019) (radio effects are present in the original voices of our corpus). They are also easier to collect since they are separated from the ambient sounds even in their final form, contrary to the dubbed film archives. Originally released in English, the game has been translated and revoiced in several other languages. In our experiments, we use the English and French versions of the audio sequences, representing about 7.5 hours of speech in each language. Segments (or recordings) are 3.5 seconds long on average. A character is then defined by a unique French-English couple of two distinct speakers. To avoid speaker identity biases, we consider only a small subset where we are certain that none of the actors play more than one character. A single audio segment corresponds to a unique speaking slot from an actor in a particular language. We then apply a filter that keeps only recordings longer than 1 second. Finally, we only keep the 16 characters for which we have the largest number of recordings, as Gresse et al. (2019) did.

In order to measure the ability of our systems to generalize, we propose a protocol that breaks down data differently. This time, we only keep 12 characters, and their 24 corresponding speakers, for the training phase and we use the remaining 4 characters for the testing phase. We still break the corpus down into training, validation and test sets. The *train* and *val* sets are composed of different recordings, but coming from the same 12 characters. To build the *train*, *val* and *test* subsets, we randomly select for each character 144, 36 and 180 recordings respectively, while balancing the number of French and English recordings. We then have a total of $1,728$ (*train*), 432 (*val*) and 720 (*test*) recordings. We name the three sets S_n.

3.1.2 SKYRIM, A SECOND VALIDATION SET TO BRING MORE VARIABILITY

Skyrim is also a known video game. We use its recordings as a second corpus in addition to the one of MassEffect previously presented. This dataset can be viewed as a validation set. We propose to use this corpus to bring out the characteristics of voices. Indeed, the validation set of MassEffect suffers from a lack of variability since it contains the same characters as the training set. Unfortunately, we do not have avaiable MassEffect data to add to the validation set. Thus, we assume that using Skyrim as a second validation set will help to bring more variability and highlight voice characteristics extracted from the p-vectors trained on MassEffect.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Curabitur pellentesque orci quis nulla tincidunt lacinia. In hac habitasse platea dictumst. Sed varius velit non justo ornare dictum non nec augue. Mauris cursus quis nulla vitae pretium. Vivamus et placerat magna. Maecenas imperdiet ipsum sed eros ultricies tempus. Nam elementum dui in nisl imperdiet congue. Fusce lacus turpis, eleifend in mauris pulvinar, euismod maximus nulla. Aliquam commodo dignissim tortor. Sed ipsum quam, commodo in dui et, consectetur mattis nibh. Vivamus finibus condimentum elit, mollis tempor nisl mollis nec. Ut et iaculis massa. Nam pellentesque nec metus nec accumsan. Etiam pulvinar iaculis risus, quis imperdiet orci placerat suscipit.

3.2 EXTRACTING VOICE REPRESENTATION FROM CHARACTER LABELS

This part of the protocol corresponds to the step 1 described in the Figure 1 of Section 2.2. In order to extract the p-vectors, we follow the best architecture described in **??**. Training this architecture need to follow two setps: 1) training a speaker identification system following a TDNN x-vector architecture. 2) replace the output softmax of this system with p-vector layers, trained on recordings of masseffect in the task of character recognition.

The x-vector model is trained on the VoxCeleb 2 corpus. The layer *XV* from which we extract the x-vectors is accompanied by a LEAKY RELU activation function and a batch normalisation. The last linear layer (*i.e.* LIN in Figure **??**) is attached to a logarithmic softmax at the output. The cost function is the cross-entropy loss. MFCCs are used as input, extracted using Kaldi tools **?** with the following parameters: 30 cepstral coefficients, 25 ms frame length, $20 - 7,600$ Hz bandwidth.

As in Gresse et al. (2020), the *DENSE* (see Figure **??**) is composed of a linear layer with hyperbolic tangent activation function and a dropout of 0.25. The p-vector layer (*i.e.* *PV* in Figure **??**) is composed of the same elements except for the dropout whose value is 0.5. We finally compute a logarithmic softmax at the output of the network. The cost function we have used to train the network is the cross-entropy loss.

3.2.1 REFINING LABELS WITH K-MEANS ALGORITHM

This part of the protocol corresponds to the step 2 described in the Figure 1 of Section 2.2.

In this work, we wish to validate our method described in 2.2. For this purpose, we focus on the k-means algorithm Unlike others, this algorithm is easier to implement. We use the implementation of the *toolkit sklearn* with the default parameters, namely: $k - means + +$ as initializer of the barycentres in order to converge more quickly to the solution. The max number of iterations is 300 of non convergence. The algorithm is runned 10 times, then we select the clustering model with the best intertia.

3.3 EXTRACTING REFINED VOICE REPRESENTATION BY USING CLUSTERING GROUPS

This part of the protocol corresponds to the step 3 described in the Figure 1 of Section 2.2.

One the clustering algorithm is complete, we build for each k-value a new neural network using the same neural architecture as used for the p-vector. Instead of predicting the initial labels, this new network is trained to predict the refined labels $r \in R$ using the clustering algorithm. After training, we can extract from its last layer a representation of the provided record that we name p-refined-vectors. The used architecture is exactly the same as for the step 1 where the neural network is firstly pre-trained on a speaker identification task by using Voxceleb 2. Only the last layer of the p-vector

network is changed to correspond to the number k of groups (of the refined labels r) instead of the number of character labels l.

3.4 EVALUATION

In this section, we present the two ways to evalute the quality of the refining. The further one is computing clustering metrics. The second one is evaluating the representation extracted using the labels on a downstream task. We give all the details on the two next subsections.

3.4.1 COMPUTING CLUSTERING METRICS

To evaluate a clustering model, different measures are used. These measures require to know for each entity in the database the group to which the algorithm has associated it as well as its original label. In this subsection, we present the following algorithms: entropy, homogeneity, completeness, v-measure, class purity, group purity and purity-K.

These measures are calculated from two different perspectives. The entropy, homogeneity and group purity measures check whether each group is associated with entities belonging to the same class. They provide a group view. The completeness and class purity measures check if each class is associated to entities belonging to the same group. They provide a class view.

To analyze the behavior of these measures, let C a clustering model learnt on the observations X with parameter k, the a priori number of clusters. Let f the function which measure the quality of the model C (e.g. entropy) on X. The table 1 shows the behaviour of the measures by computing for each of them the limit of f i $k \to +\infty$ et $k \to 0$.

	angle de vue	$\lim_{k\to 1} f(C;X)$	$\lim_{k\to \lvert X\rvert} f(C;X)$
entropy	group	$+\infty$	0
homogeneity	group	0	1
group purity	group	0	1
completeness	class	1	0
class purity	class	1	0
v-measure	hybrid	$\times$	$\times$
purity-K	hybrid	$\times$	$\times$

Table 1: Functional study of clustering measures

These measures all have their meaning and are generally misinterpreted. This is particularly the case for the entropy applied to clustering algorithms. The latter is a measure similar to that of homogeneity that has been inverted. Although it helps to compare two systems, it has a major drawback. It is bounded between 0 and $+\infty$. Interpreting the measure of a single model is therefore difficult unlike homogeneity which is bounded between 0 and 1. The latter thus gives a rate of homogeneity that entropy cannot give. Without going into details here, the group purity is a measure very similar to the homogeneity.

The measures of completeness and class purity, in addition to being very similar, are the antithesis of the previous measures. They are no longer measures of group homogeneity but measures of class homogeneity. The closer the value is to 1, the more the class in question is observed in only one group. The closer the value is to 0, the more the class in question is observed in a large number of groups at similar quantities.

Hybrid measures are weighted averages of class and group measures. Their limitations in $k \to +\infty$ and $k \to 0$ are not clear as they depend on the balance of power between homogeneity (group purity) and completeness (class purity).

Demonstrating the limits of each function is quite simple. For homogeneity and group purity, it is maximal when the elements of each group belong to the same class. When we reach$k = |X|$, each element is then assigned to a cluster and each cluster is assigned to an element. There is then only one class for each group. The homogeneity is thus maximal, the group purity too. In the opposite case, when $k = 1$, all observations belong to the same group. The group is then maximally heterogeneous

and the values of homogeneity and group purity then tend towards 0. For completeness and class purity, the reasoning is similar and then becomes trivial.

3.4.2 EVALUATING VOICE REPRESENTATION

As described in Quillot et al. (2021), we evaluate p-vectors on a character voice similarity task using a character-similarity measure based on a Siamese network. We generate *Target* trials composed of pairs of recordings belonging to the same character and *Non-target* trials made up of recordings belonging to two different characters. To avoid any bias, the number of *targets* and *non-targets* is balanced, as well as the number of pairs between two actors. So, we generate $124,416$ (*train*), $7,776$ (*val*) and $64,800$ (*test*) *trials*. The threshold is set *a posteriori* at the Equal Error Rate (EER) point. System performance is then expressed in terms of accuracy on the test.

4 EXPERIMENTS AND RESULTS

In the experiments we present in the following sections, we decided to test different values of k in order to limit the number of calculations to be performed. Several values seemed obvious to test such as 12, 24, 4 and 8 which are the number of characters and speakers respectively in the train and test. We also assume that the number of speech features we are looking for is higher than the number of characters our system has to learn to model. We therefore also tested the values 18, 32, 48 and 64 which are arbitrary values but greater than 12. On the other hand, we have also tested $k = 2$ and $k = 6$ as values less than 4 and 12 as a counter-hypothesis.

4.1 REFINING P-VECTORS BY USING MASSEFFECT

In the Table 2, we plot the observed metrics *v-measure* and *purity-K* on our clustering models computed on the training data of S_n (MassEffect). We then train 10 siamese neural networks and select the best on the validation. Its accuracy sore on the validation and the test are plotted on the two last lines. For the three metrics presented in the table, the higher the value, the better the system.

On the validation set, the metrics (v-measure and purity-K) highlight the 12 characters contained in them and known to the system. We naturally have higher values for the value $k = 12$ with 0.91 for the v-measure and 0.90 for the purity-K.

On the test, we observe a somewhat different phenomenon. Remember that the test is composed of 4 characters. If the points – which are projections of the records into p-vector space – were perfectly discriminated by the neural classification system, they would then be organized into groups of records belonging to the same character. We would therefore have stronger measure scores for $k = 4$. In our case, we can observe this phenomenon with a loss due to the difficulty for the system to separate the records of characters that are unknown to it. Indeed, the highest values are at $k = 6$ for the v-measure with 0.58 and at $k = 2$ for the purity with 0.70. These groups are finally the closest to 4 and show a loss of quality of the representations on unknown data. However, they do not show, a priori, the existence of features that could interest us.

Finally, we can be interested in the performances obtained on the Siamese networks that we have trained. Recall that p-refined-vectors are provided as input to these Siamese networks which are trained on a character verification task. These vectors are extracted from a neural network trained to recognize the clustering group (refined label) corresponding the given recording (observation). Without surprise, we observe a best accuracy for $k = 12$. This corresponds to the number of characters. Contrary to the previous metrics, we do not observe a normal distribution in the results. Indeed, when k is moving away from 12, the obtained accuracy is not necessarily lower. For instance, at $k = 64$ the obtained accuracy is not significantly lower than the one obtained at $k = 12$. Even more interesting, the accuracy obtained on the test subset is about 0.70 that is higher than the state of the art system (< 0.69 Quillot et al. (2021)).

4.2 USING SKYRIM AS A SECOND CORPORA

In a second experiment, we wish to verify if it is possible to use a secondary corpus to extract voice features. For this purpose, we propose this time to apply the clustering algorithm on the p-vectors

	2	4	6	8	12	18	24	32	48	64
v-measure val	0.40	0.62	0.73	0.80	**0,91**	$0,86$	$0,83$	$0,80$	$0,75$	$0,74$
v-measure test	0.54	0.56	**0.58**	0.50	0.48	0.45	0.44	0.44	0.41	0.41
purity-K val	0.41	0.53	0.64	0.73	**0.90**	0.78	0.68	0.61	0.52	0.48
purity-K test	**0.70**	0.65	0.64	0.54	0.50	0.45	0.43	0.40	0.32	0.31
siamese val	0.67	0.73	0.80	0.87	**0.94**	$0,92$	$0,93$	0.93	$0,92$	**0,93**
siamese test	0.55	0.65	0.62	0.66	0.66	0.69	0.63	0.66	0.63	**0.70**

Table 2: Results obtained by using the recordings of MassEffect for the clustering algorithm, k-means. The v-measure and purity measures are computed on the clusters extracted from the p-vectors. The computed accuracy ("siamese val" and "test" lines) are obtained on the siamese networks trained with the refined p-vectors.

representations extracted from the Skyrim corpus for step 2. The p-vectors have learned to highlight the played character but it is impossible to bring out the vocal characteristics it extracts to perform this classification. We hope that the use of a secondary corpus such as Skyrim will create enough variability to bring out the initial labels for vocal features. For step 1, the p-vector extractor is exactly the same as the previous experiment. It is therefore trained on MassEffect with S_n and the Skyrim data is completely unknown to it. For step 2, we extract p-vectors using Skyrim records and perform on them the k-means algorithm using the same values for k. For step 3, we finally associate a cluster to each record of MassEffect and train p-refined-vectors as realized in the previous experiment.

Table 3 presents the obtained results. We can observe that this time the v-measure and the purity-K move away from the number of characters that we originally had. So we know that the semantics of these clusters differ from the semantics of the character labels. In this example, purity-K highlights $k = 2$. In the $k = 2$ case, the two groupings primarily separate men and women. Each group is therefore the gender characteristic of the voice. For the v-measure, the score correlates with the number k of groups. The best score is therefore for $k = 64$, but it might have gone up by increasing k. Unlike $k = 2$, it is difficult to understand the semantics of each group and to associate words that make sense to a human operator. Note that these measures have no known bias unlike the previous experiment since here the number of characters in the Skyrim data is 32 and the $k = 2$ and $k = 64$ values are significantly off.

	2	4	6	8	12	18	24	32	48	64
v-measure	0.23	0.27	0.30	0.33	0.36	0.37	0.38	0.40	0.41	**0.42**
purity	**0.26**	0.23	0.23	0.24	0.24	0.24	0.23	0.23	0.22	0.22
siamese val	0.67	0.72	0.78	0.84	0.87	0.90	0.90	0.90	**0.91**	$\times$
siamese test	0.50	**0.72**	0.65	0.63	0.67	0.62	0.66	0.66	0.62	$\times$

Table 3: Results obtained using Skyrim records for the k-means clustering algorithm. The v-measure and purity-K measures are computed on the clusters extracted from the p-vectors. The success rate measures computed (lines "siamese val" and "test") are obtained on the siamese networks trained with the refined p-vectors.

4.3 Assessing the impact of the distance selected for k-means

As we know, the role of distance in the k-means algorithm is very important. For this reason, we propose to realise the same experiment as presented in the previous section (4.2) but using *cosine* and *mahalanobis* distances instead of the euclidian one.

The table 4 presents obtained results by using the cosine and the mahalanobis distances. We can observe that the Mahalanobis distance offers superior results on v-measure and purity-K compared to the Euclidean distance and the cosine distance. Moreover, while the cosine distance tends to draw the same conclusions as the Euclidean distance through these measurements, the mahalnobis distance differs in the number of clusters it highlights with the purity-K measure. Indeed, the two other distances put forward $k = 2$ while this one puts forward $k = 32$. The training of Siamese networks seems to agree with the choice $k = 32$ for mahalnobis. However, its results contrary to the cosine distance are significantly lower with 0.64 against 0.67.

	2	4	6	8	12	18	24	32	48	64
distance cosine										
v-measure	0.23	0.27	0.30	0.32	0.35	0.37	0.38	0.39	0.41	**0.41**
purity-K	**0.25**	0.24	0.23	0.24	0.24	0.24	0.23	0.23	0.23	0.21
siamese val	0.70	0.71	0.78	0.84	0.88	**0.91**	0.90	0.90	×	×
siamese test	0.56	**0.70**	0.63	0.63	0.66	0.67	0.69	0.66	×	×
distance de mahalanobis										
v-measure	0.22	0.30	0.31	0.36	0.40	0.43	0.47	0.52	0.54	**0.55**
purity-K	0.26	0.25	0.24	0.26	0.29	0.30	0.32	**0.36**	0.36	0.35
siamese val	0.63	0.69	0.74	0.77	0.77	0.78	0.80	**0.80**	×	×
siamese test	0.59	0.59	0.56	0.61	0.54	0.45	0.55	**0.64**	×	×

Table 4: With Skyrim and cosine distance and mahalanobis. Results obtained using Skyrim records for the k-means clustering algorithm and cosine and mahalanobis distances. The v-measure and purity-K measures are computed on the clusters extracted from the p-vectors. The computed success rates (line "siamese val" and "test") are obtained on the Siamese networks trained with the refined p-vectors.

5 Conclusion

References

Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives, 2014.

Adrien Gresse, Mathias Quillot, Richard Dufour, Vincent Labatut, and Jean-François Bonastre. Similarity metric based on siamese neural networks for voice casting. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019.

Adrien Gresse, Mathias Quillot, Richard Dufour, and Jean-Francçois Bonastre. Learning Voice Representation Using Knowledge Distillation For Automatic Voice Casting. In *Interspeech*, 2020.

Alexander Heimerl, Tobias Baur, Florian Lingenfelser, Johannes Wagner, and Elisabeth André. Nova - a tool for explainable cooperative machine learning. In *2019 8th International Conference on Affective Computing and Intelligent Interaction (ACII)*, pp. 109–115, 2019. doi: 10.1109/ACII.2019.8925519.

Mathias Quillot, Jarod Duret, Richard Dufour, Mickael Rouvier, and Jean-François Bonastre. Influence of speaker pre-training on character voice representation. In Alexey Karpov and Rodmonga Potapova (eds.), *Speech and Computer*, pp. 577–588, Cham, 2021. Springer International Publishing.

Vassil Roussev. An evaluation of forensic similarity hashes. *Digital Investigation*, 8:S34–S41, 2011. ISSN 1742-2876. doi: https://doi.org/10.1016/j.diin.2011.05.005. URL `https://www.sciencedirect.com/science/article/pii/S1742287611000296`. The Proceedings of the Eleventh Annual DFRWS Conference.