Statistical Scoring for a D-vector Speaker Recognition System

Kenneth Li

Summer 2018 @Center for Speech and Language Technology
Tsinghua University, Beijing, China
July 19, 2018

kenneth.j.li@vanderbilt.edu

Abstract

Feature-based speaker recognition systems have traditionally used the simplistic cosine similarity approach for scoring. To compete with end-to-end architectures, more complex metrics leveraging discarded information like variance in data can improve this scoring task using statistical distances. This paper explores the model-free usage of Z-scores, Kullback-Leibler divergence, and other factors to reduce EER of a baseline scoring backend. Optimization of simple linear factor combinations yield improvements of up to 0.5% from baseline yet solutions are unstable and benefits can diminish with large trial sizes. These require more work for practical value. Model-based usage of statistical features can be promising with significant EER reduction.

1 Introduction

A text-independent speaker verification system based on convolutional time-delay deep neural networks [3] has been shown to extract valuable features (d-vectors) from utterances so speakers could be verified without an end-to-end architecture. Although these learned features are versatile for future exploration, their verification performance in terms of equal error rate (EER) can use improvement. The traditional method for scoring similarity of two speakers consists of averaging all respective frame-level d-vectors by utterance, averaging these results by speaker, and then taking the cosine distance between the final two vectors. This approach is simple and effective, but does not take into account additional information, like variance between utterance or frame-level vectors.

Past approaches to lowering EER of other systems that produce learned features include PLDA variants and similarity learning. Lei et al.[2] makes use of the Mahalanobis metric to measure statistical distance between i-vectors of speakers to increase performance of their scoring backend. While this approach has been shown to be effective, there is motivation to explore model-free metrics for a more simplistic scoring backend.

The goal of this project was to identify other metric combinations that would result in a lower EER compared with the cosine similarity baseline when leveraging additional information like variance. Ideally, these metrics would be easy to compute and would not involve additional machine learning for sake of simplicity in the scoring backend of this system. This approach takes several other statistics-based metrics to form a linear combination with the baseline cosine metric and improve EER. This is a simple way of incorporating additional information and observing results.

With small samples, it was found that these factors made significant reductions of about 0.5% to the baseline EER. Machine learning is also briefly experimented with to explore the performance and the possibility of learning metrics with a model-based scoring backend.

Section 2 describes the proposed distances and methods implemented in this project. Section 3 more explicitly details these methods, displays results of experiments, and discloses motivations of certain procedures. Section 4 concludes these findings and discusses future work.

2 Methods

2.1 Testing platform

Another focus of this project was to develop a robust foundation for running experiments as the nature of this research beckons quantity over quality. Multiple factors were tested in rapid succession over a dataset 10's of GBs large. Since the memory capacity of available machines were relatively small, practices with HDF5 files, sequential load-

ing, and dataset sampling were generously deployed. To run experiments, the Dask multiprocessing library was used to execute Python code in parallel and improved processing speeds by up to 500%. The bulk of this project was actually spent on pre-processing data and testing the full capabilities of this library which has yet to gain popular feedback.

2.2 Factors

Speaker data is represented in the following fashion: each line in a list of independent trials consists of an enrolled speaker, a test speaker, and a binary class indicating whether the speakers were the same person or not. Each enrolled speaker had 5 unique utterances, and each utterance had a variable number of 400-length d-vectors. The case for test speakers is identical but each had 3 unique utterances. The explored factors that appeared valuable are listed below.

- **Z-Score**: Given a set of enrolled speaker vectors and a single test speaker vector (usually the mean of speaker-level vectors), get the column-wise Z-score vector. Each test vector value is used as the test against the sample distribution of enrolled vector values in the same column. This produces a 400-length vector of 'z-scores' in this case, and the mean of absolute values is taken to produce a single score for each trial pair of speakers. In theory, a larger score would indicate greater dissimilarity between speakers. This factor can also be applied to frame-level vectors.

- **KL Divergence**: Kullback-Leibler divergence is known to measure similarity between two distributions. Although this is not a true distance metric, we can use this measure in a crude way to gauge its potential on this data. A speaker-level "KL-score" is computed as follows. Since enrolled speakers had 5 utterances and test speakers had 3 utterances in this dataset, the 3 test utterances were used as a "sliding window" over the 5 enrolled utterances such that 3 KL-divergence vectors would be obtained by comparing each pair of 3 utterances in each category. The mean of each vector is taken, and the minimum is returned for each pair of speakers. This factor can also be applied to frame-level vectors.

- **Manhattan Distance**: Take the mean of all possible combinations of enroll-test speaker-level vector pairs. In this data, there would be 15 such pairs from 5 utterances per enrolled speaker and 3 utterances per test speaker.

Additional consideration included **Mahalanobis distance**, and just about every other statistical distance [4]. The former was tested from a variety of angles, but did not yield consistent results because the data was too high-dimensional. Even after dimension reduction with PCA, Mahalanobis distance still failed to be a valuable factor on its own. Other statistical distances were not implemented due to lack of time or my belief that they would all yield very similar results to aforementioned factors.

2.3 Common ML Models

As an addendum, simple features can be taken from a pair of speakers and their vector data to create a model-based classification task based on whether the pair was a match or not. Factors may include cosine similarity between speakers and variance between rows of average utterance vectors. Common models like SVM and Random Forest were trained on these features and EER was evaluated based on their predictions on test data. Although this approach deviated from the goal of the project, the Random Forest model was found to perform significantly better in reducing EER than the non-model-based method above.

3 Experiments

Data is produced from the open speech database THCHS-30 [1]. There are 11725308 total trials, with 17892 positive classes, or about 0.15% of the total. Since high accuracy can be trivially obtained, EER is a more valuable performance metric.

3.1 Individual Factor Performance

Speaker-level			
EER	Factor	Sample Size	Compute Time
7.897	Cosine	1	10m
7.963	Cosine	0.5	5m
10.586	KL-Div	0.5	31m
21.466	Z-Score	0.5	33m
10.528	Manhattan	0.5	17m

Table 1. EER of individual factors

Results of frame-level experiments were discarded because they performed significantly worse than their speaker-level counterparts. For 100k samples, KL Divergence on frame-level data yielded an EER of 15%, as opposed to the 10% shown in Table 1. Z-score on frame-level data yielded an EER upwards of 40% on an equally smaller sample size. As these experiments also took significantly longer to compute, usage of frame-level data on large sample sizes was disregarded.

3.2 Factor Combination Performance

Optimization was run for the simple linear combination with minimal EER:

$$\alpha_1 Cos + ... + \alpha_{i-1} Factor_{i-1} + \alpha_i Factor_i \quad (1)$$

The following is a list of experimental setups.

1. **CosZKL-F-100k**: On a sample size of 100k trials, speaker-level cosine score is complemented with speaker-level Z-score and *frame-level* KL divergence.
2. **CosZKL-100k**: Same as CosZKL but all factors are speaker-level.
3. **CosZKL-1m**: Same as CosZKL-100k but on a 10% size sample of the full trials set.
4. **CosZKLM-1m**: Speaker-level cosine similarity, Z-score, KL divergence, and Manhattan distance on 10% sample.
5. **ZKLM-1m**: Same as CosZKLM-1m but the scalar to cosine scores is a fixed constant of 100.
6. **CosZKLM-5m**: Same as CosZKLM-1m but on a 50% sample.

Results displayed are from Powell's method of optimization since these yielded the lowest EER with a large degree of consistency. A starting guess of 1 for cosine similarity and -0.5 for all other factors was used for all setups to be consistent and this oftentimes yielded the best minimization results, which are reported in the table below. However, it should be noted that other guesses may yield better results for certain setups.

Opt EER	Base EER	Setup	Solutions
8.247	9.104	CosZKL-F-100k	[108, -0.49, -0.11]
7.299	9.104	CosZKL-100k	[6.6, 0.10, -282.9]
8.000	8.507	CosZKL-1m	[46.3, -0.27, -189]
7.849	8.507	CosZKLM-1m	[113,-0.6, -3.8, 0.36]
7.932	8.507	ZKLM-1m	[-0.68, 3.02, 0.21]
7.943	7.963	CosZKLM-5m	[5.68, 77.2, 0.036, -0.023]

Table 2. Results of optimized factor combinations

While these scalar solutions shown in the above table were found to minimize EER, it is noted that such solutions can widely vary. On larger samples, it was found that EER improvements from optimization were diminished significantly. The different results between CosZKL-F-100k and CosZKL-100k suggest that using frame-level data is overly noisy and obtrusive to prediction. In many of these trials, cosine remains as the 'dominant' factor with the largest scalar solution, which reflects results of Section 3.1. Different factors or functions should be explored to produce a practical result.

3.3 ML Model Performance

A few experiments making explicit use of common classification models were conducted and detailed below. These models primarily made use of features like cosine similarity and variance between enrolled speaker-level vectors.

1. **CosVar-SVM**: A scalar cosine similarity score is append to a 400-length variance vector computed by taking the column-wise variance of 5 speaker-level enrollment vectors. Each trial then has 401 features from these factors. An SVM model with a linear kernel is trained using this data and trial classes.
2. **CosVar-RF**: A Random Forest classifier is trained using the same data as CosVar-SVM.
3. **Cos-RF**: A Random Forest classifier is trained using just cosine similarity as a feature.

Results are displayed below. The first instance of each setup indicate the training set, and all instances indicate prediction results.

Pred EER	Base EER	Setup	Sample Size
40.620	7.061	CosVar-SVM	0.01
0.033	7.061	CosVar-RF	0.01
0.229	8.507	CosVar-RF	0.10
0.055	7.061	Cos-RF	0.01
0.257	8.507	Cos-RF	0.10

Table 3. Results of trained models

It is almost comical and slightly alarming how the performance of these models is so drastically dialed-up. While ML models may provide huge performance value, the finer relationships within the d-vector data are abstracted. A model may not provide as much insight into the nature of extracted features than straightforward distance metrics. The future usage of an ML setup is up to the discretion of system maintainers. The RF classifier took about 7 seconds to train on a 0.01% size sample using CosVar features with

8 parallel jobs. While other factors like overfitting need to be considered to scale a model, the RF model may be a candidate for a better scoring backend. Feature importance can also be further explored in the random forest model.

4 Conclusion

Despite the intuition that factoring additional information can reduce EER of this speaker system's scoring backend, there is more research to be done on realizing the potential of this claim as results demonstrate. Although EER of individual proposed factors was poor, optimized factor combinations can yield slight improvements to baseline EER. Yet, these solutions can widely vary and such a method is in need of more work before practical usage. The usage and implementation of researched factors can be significantly refined from their presented forms to boost performance. There are an innumerable number of metrics to experiment with and Sections 3.1, 3.2 suggest that the relationship between variance and EER is nonlinear and nontrivial. Because this interaction is difficult to precisely characterize without rigorous mathematics, a model-based scoring backend may be fit for further development.

5 Future work

Future work may include deeper analysis of the data itself to pave way for new factors better than traditional cosine similarity. Additionally, more complex factor combinations can be explored to yield more consistent solutions to optimized EER results. Metric and similarity learning are also potential techniques to directly learn more valuable distances. Closer analysis of the models briefly used can also lend further insight into the nature of d-vectors.

6 Acknowledgements

Thank you to Professor Dong Wang for the opportunity to work at CSLT on this project and the guidance to conduct capable research. Thank you to Dr. Zhiyuan Tang for continuous feedback on my progress and encouragement on results.

References

[1] Z. Z. Dong Wang, Xuewei Zhang. Thchs-30 : A free chinese speech corpus, 2015.

[2] Z. Lei, Y. Wan, J. Luo, and Y. Yang. Mahalanobis metric scoring learned from weighted pairwise constraints in i-vector speaker recognition system. In *Interspeech 2016*, pages 1815–1819, 2016.

[3] L. Li, Y. Chen, Y. Shi, Z. Tang, and D. Wang. Deep speaker feature learning for text-independent speaker verification. *CoRR*, abs/1705.03670, 2017.

[4] Wikipedia contributors. Statistical distance — Wikipedia, the free encyclopedia. `https://en.wikipedia.org/w/index.php?title=Statistical_distance&oldid=843427668`, 2018. [Online; accessed 19-July-2018].