Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2019) 27: 3758 – 3762
© TÜBİTAK
doi:10.3906/elk-1901-231

Scale-invariant MFCCs for speech/speaker recognition

Zekeriya TÜFEKCİ[1], **Gökay DİŞKEN**[2,*]
[1]Department of Computer Engineering, Faculty of Engineering, Çukurova University, Adana, Turkey
[2]Department of Electrical and Electronics Engineering,
Faculty of Engineering, Adana Science and Technology University, Adana, Turkey

Received: 31.01.2019 • **Accepted/Published Online:** 24.05.2019 • **Final Version:** 18.09.2019

Abstract: The feature extraction process is a fundamental part of speech processing. Mel frequency cepstral coefficients (MFCCs) are the most commonly used feature types in the speech/speaker recognition literature. However, the MFCC framework may face numerical issues or dynamic range problems, which decreases their performance. A practical solution to these problems is adding a constant to filter-bank magnitudes before log compression, thus violating the scale-invariant property. In this work, a magnitude normalization and a multiplication constant are introduced to make the MFCCs scale-invariant and to avoid dynamic range expansion of nonspeech frames. Speaker verification experiments are conducted to show the effectiveness of the proposed scheme.

Key words: Feature extraction, speaker recognition, speech recognition

1. Introduction

Feature extraction is one of the main processes in automatic speech/speaker recognition. Despite numerous feature types found in the literature, spectral-based techniques such as mel frequency cepstral coefficients (MFCCs) [1] remain popular due to their ease of implementation and high performance in clean conditions. Conventional MFCC extraction schemes include windowing speech frames, taking their Fourier transform, filtering the spectrum with a filter bank linearly spaced in mel-scale, log compression, and taking the discrete cosine transform. MFCCs are scale-invariant except the zeroth coefficient, which is related to the signal energy. However, if a filter bank magnitude is zero, the logarithm goes to negative infinity, hence causing numerical problems. On the other hand, if filter bank magnitudes are close to zero, the dynamic range of the log compressed values increases, which decreases the recognition performance.

To overcome this issue, several methods have been proposed by researchers, such as adding a constant value (i.e. one) to filter bank magnitudes as in the HTK toolkit [2], replacing log compression with root compression [3], and dithering the clean signal by adding a small amount of noise [4]. Although these methods increase the recognition performance, they also introduce some handicaps. Speech signals with low magnitudes may be lost in the noise in the dithering. Cepstral mean normalization cannot be used with root compression. Adding a constant value to filter bank magnitudes or a ceiling to keep the smaller values to a constant before log operation violates the scale-invariant property of MFCCs.

In this work, we focused on making the MFCCs completely scale-invariant by adding a magnitude normalization scheme to the filter bank magnitudes. Previously, dynamic range normalization was proposed

*Correspondence: gdisken@adanabtu.edu.tr only for the zeroth coefficient [5]. In this work our main concern is the coefficients other than the zeroth. There are many energy normalization techniques in the literature, such as in [6] and [7]; however, it is unclear whether a normalization step is added in many speech/speaker recognition studies. We utilized a recently proposed voice activity detection algorithm [8] to detect speech regions in each filter of the filter bank separately and apply energy normalization based on these regions.

2. Normalization and scaling

The conventional MFCC extraction scheme includes a logarithmic compression step, as given by Eq. (1):

$$M = log(x), \tag{1}$$

where x denotes the filter bank magnitude and M denotes its compressed form. Eq. (1) has two problems. First, if x is zero, the logarithm goes to negative infinity, hence causing numerical problems. Second, Eq. (1) will compress big values of x, which represent the speech signal, and will expand the small values of x (i.e. $x << 1$), which represent the nonspeech signal. Expanding the dynamic range of the nonspeech parts of utterances may degrade the recognition performance of the systems.

To overcome the numerical problems, a constant (usually 1) is added to x as given in Eq. (2):

$$M = log(1 + x). \tag{2}$$

However, Eq. (2) does not offer a solution to the second problem since $log(1 + x)$ will generate the same values as Eq. (1) if the average value of x is sufficiently big. In addition to this, Eq. (2) creates two new problems. First, it may create a mismatch between training and test signals, because $log(1 + x)$ is not scale-invariant. MFCCs for the same word recorded with different volume levels will vary. MFCCs obtained using $log(1 + x)$ will be different than the ones obtained by $log(1 + ax)$ where x is the original signal and ax is the scaled version. The second problem is that for the small values (i.e. $x << 1$) the log compression loses its effect since $log(1 + x) = x$.

The result and hence the effectiveness of the log operation depend on the average of the filter bank magnitudes. This explains why MFCCs are not really scale-invariant. To make the MFCCs scale-invariant and reduce the deteriorative effects of the log operation, a constant multiplier (c) to the normalized filter bank magnitudes is introduced in this work as in $log(1+c(x/\hat{x}))$, where $\hat{x}$ is the average filter bank magnitude for the given filter bank over all speech frames. Dividing x by $\hat{x}$ normalizes the filter bank magnitudes. The dynamic range expansion of the nonspeech parts of the utterances will be avoided by choosing an optimum c value.

The constant multiplier and magnitude normalization offer a straightforward solution to the aforementioned problems. The crucial point is normalizing the filter-bank magnitude values. In this work, a recently proposed [8] voice activity detector is utilized to detect speech regions for the normalization. For each filter, average speech magnitude is calculated by using these regions. Finally, filter bank magnitudes of the given band are divided into its average magnitude to achieve magnitude normalization.

It should be noted that the normalization is done to the entire utterance, not only the detected speech-dominant frames. The reason is that there is no ideal voice activity detector (VAD), so silence/noisy frames may be misclassified as speech frames. Along with this, to show that the proposed multiplication effectively avoids the dynamic range problem mentioned before, all utterances (training and testing) are used as a whole (i.e. speech/silence regions were not separated).

Table 1. Baseline results (EERs) for conventional MFCC features.

		3 s	10 s	30 s
GMM-UBM	log(1+X)	9.32	5.42	4.20
	log(X)	9.86	5.65	3.74
i-vector	log(1+X)	11.31	5.81	3.21
	log(X)	11.39	5.50	3.36

3. Experiments

The proposed modification is tested in text-independent speaker verification experiments using the male portion of the NIST SRE 1998 database. A total of 250 male speakers with training data of about 5 min for each speaker are available. Test data durations are 3, 10, and 30 s. There are 1308 test speech files for each duration. The extracted features are 13 MFCCs, excluding the zeroth coefficient, and their deltas. For the back end, the conventional GMM-UBM system and the more recent i-vectors are used. A 1024-mixtures UBM is trained by pooling all training data. The speaker models are then adapted from the UBM with a relevance factor of 16. The same UBM is used in the i-vector extraction scheme, which also includes linear discriminant analysis and PLDA scoring. The toolbox provided in [9] is used for all of the back end processes.

Before providing the results for the proposed method, the baseline scores are given in Table 1 for conventional MFCCs with $log(X)$ and $log(1+X)$. X indicates the traditional features without any normalization.

Table 2 shows the equal error rates (EERs) obtained with the conventional GMM-UBM back end. Note that x denotes the normalized filter bank magnitude, as explained previously. The percent increase in the relative error for the given c value is calculated as:

$$E_c = \frac{EER_c - EER_b}{EER_b} \times 100, \tag{3}$$

where EER_c is the EER for the given test duration and c. EER_b is the smallest EER for each test data duration (e.g., 7.18 for 3 s, 4.28 for 10 s, and 3.21 for 30 s in Table 2). E_c is the relative change for the given c value against the best case. EER_b is used to examine the relative change between the results of different c values. This relative change may indicate trends towards an optimal c value. In practice, the length of an incoming utterance may vary, and hence it is more desirable to find the c value that can show sufficient performance under different data lengths. Therefore, the average of the relative changes are calculated and given in the last column of Table 2 as percent increase in the relative error for the given c value. The performance of the system decreases as the c values move away from an optimal area, as expected.

Similarly, Table 3 shows the EERs obtained with the state-of-the-art i-vector back end. Column information is the same as in the previous table. Compared to the baseline results, the proposed method achieved better performances for several c values around an optimal point.

For the small multiplication constants, the negative effects of the lower magnitudes may not be eliminated. As the constants increased, the lower magnitudes successfully moved to a range in which the log operation can effectively compress them.

Similar results can be obtained from both tables. This situation confirms that the performance of the MFCCs can be enhanced with the proposed modification, regardless of the back end used in the system. The best performance for the i-vector method is achieved when the multiplier is 200. For the GMM-UBM method, the best performance is yielded with 300.

Table 2. Verification results (EERs) for different c values with GMM-UBM back end.

log(1+x*c)	3 s	10 s	30 s	Average E_c
log(1+x*1)	19.34	12.99	7.64	170.27
log(1+x*10)	14.29	8.86	5.81	95.67
log(1+x*20)	11.54	7.18	4.81	59.36
log(1+x*30)	10.39	6.49	4.28	43.15
log(1+x*40)	9.48	5.88	3.9	30.23
log(1+x*50)	9.09	5.42	3.59	21.62
log(1+x*100)	8.02	4.6	3.36	8.46
log(1+x*200)	7.49	4.43	3.28	3.40
log(1+x*300)	7.41	4.28	3.21	1.06
log(1+x*400)	7.34	4.28	3.28	1.50
log(1+x*500)	7.18	4.43	3.36	2.77
log(1+x*600)	7.35	4.35	3.36	2.92
log(1+x*700)	7.35	4.43	3.44	4.28
log(1+x*800)	7.49	4.43	3.36	4.19
log(1+x*900)	7.26	4.51	3.44	4.52
log(1+x*1000)	7.41	4.58	3.51	6.62
log(1+x*2000)	7.41	4.51	3.59	6.81
log(1+x*2500)	7.33	4.58	3.59	7.05
log(1+x*10000)	7.26	4.58	3.66	7.49

Table 3. Verification results (EERs) for different c values with i-vector back end.

log(1+x*c)	3 s	10 s	30 s	Average E_c
log(1+x*1)	12.31	5.5	3.44	51.94
log(1+x*10)	9.93	4.51	2.82	23.84
log(1+x*20)	10.85	4.58	2.9	29.69
log(1+x*30)	10.93	4.58	2.6	26.08
log(1+x*40)	10.78	4.58	2.6	25.41
log(1+x*50)	10.85	4.58	2.6	25.72
log(1+x*100)	8.02	3.89	2.67	7.51
log(1+x*200)	7.79	3.59	2.6	2.76
log(1+x*300)	7.41	3.89	2.9	7.81
log(1+x*400)	7.95	3.97	2.75	8.99
log(1+x*500)	8.02	3.74	2.9	9.16
log(1+x*600)	8.1	4.05	2.82	11.34
log(1+x*700)	7.87	4.05	2.75	9.38
log(1+x*800)	7.95	4.2	2.82	12.06
log(1+x*900)	7.79	4.12	2.52	6.63
log(1+x*1000)	9.32	4.96	2.75	24.35
log(1+x*2000)	9.71	4.74	2.6	22.08
log(1+x*2500)	7.56	3.82	2.82	6.77
log(1+x*10000)	9.32	4.74	2.52	19.26

For small constants, log operation loses its compression ability. Therefore, the performance degrades drastically. On the other hand, when the constant becomes much bigger than its optimum value, log operation increases the dynamic range of nonspeech parts. However, the performance drop is not dramatic as in the small constant cases. The performance is more sensitive to the smaller constants than the bigger ones, relative to the optimum value of the constant as seen in the tables. It should also be noted that the constant 1 is added to avoid numerical problems in the event that a zero magnitude occurs somehow.

4. Conclusion

In this study, a modification to the conventional MFCC extraction scheme was proposed. The main purpose of this scheme is to overcome a fundamental problem in the log compression step, which degrades the recognition performance. The log operation compresses the frames with high magnitudes/energies. On the other hand, it expands the range of noisy/low-energy frames, hence creating a dynamic range expansion, or numerical problems. This unintentional result especially occurs in practical applications.

The proposed modification consists of magnitude normalization, followed by multiplication with a constant. Normalization was utilized to reduce the magnitude variation between the utterances. The constant multiplication is applied to the normalized filter bank outputs in order to scale them into a range where the log operation can effectively compress them. By this scaling, the mismatch due to the expanded range of the low magnitude frames is easily reduced.

The experiments showed that the proposed modification is effective for both GMM-UBM and i-vector methods. Similar performance progress can be observed for both modeling methods. Compared to the conventional MFCCs, the proposed method offers simple yet effective performance increments. This could be support for choosing the modified MFCCs over the conventional ones.

References

[1] Davis SB, Mermelstein P. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. IEEE Transactions on Acoustics, Speech, and Signal Processing 1980; 4: 357-366.

[2] Young S, Kershaw J, Odell D, Valtchey V, Woodland P. The HTK Book Version 3.0. Cambridge, UK: Cambridge University Press, 2000.

[3] Alam MJ, Kenny P, O'Shaughnessy D. Robust feature extraction based on an asymmetric level-dependent auditory filterbank and a subband spectrum enhancement technique. Digital Signal Processing 2014; 29 (1): 147–157.

[4] Borsky M, Mizera P, Pollak P, Nouza J. Dithering techniques in automatic recognition of speech corrupted by MP3 compression: Analysis, solutions and experiments. Speech Communuication 2017; 86 (1): 75–84. doi: 10.1016/j.specom.2016.11.007

[5] Zhu W, O'Shaughnessy D. Log-energy dynamic range normalization for robust speech recognition. In: IEEE International Conference on Acoustics, Speech, and Signal Processing; Philadelphia, PA, USA; 2005. pp. 245–248.

[6] Giurgiu M, Kabir A. Improving automatic speech recognition in noise by energy normalization and signal resynthesis. In: IEEE 7th International Conference on Intelligent Computer Communication and Processing; Philadelphia, PA, USA; 2011. pp. 311–314.

[7] Li Q, Zheng J, Tsai A, Zhou Q. Robust endpoint detection and energy normalization for real-time speech and speaker recognition. IEEE Transactions on Speech and Audio Processing 2002; 10 (3): 146–157.

[8] Dişken G, Tüfekci Z, Çevik U. A robust polynomial regression-based voice activity detector for speaker verification. EURASIP Journal on Audio, Speech and Music Processing 2017; 2017 (23): 1-16.

[9] Sadjadi SO, Slaney M, Heck L. MSR Identity Toolbox v1.0: A MATLAB toolbox for speaker recognition research. IEEE Speech and Language Processing Technical Committee Newsletter 2013; 1 (4): 1–4.