A Novel Learnable Dictionary Encoding Layer for End-to-End Language Identification

Weicheng Cai[1], *Zexin Cai*[1], *Xiang Zhang*[3], *Xiaoqi Wang*[4] and *Ming Li*[1,2]

1. School of Electronics and Information Technology, Sun Yat-sen University, Guangzhou, China
2. Data Science Research Center, Duke Kunshan University, Kunshan, China
3. Tencent inc., Bejing, China
4. Jiangsu Jinling Science and Technology Group Limited, Nanjing, China

ml442@duke.edu

Introduction

In recent decades, in order to get the utterance level vector representation, dictionary learning procedure is widely used.

A dictionary, which contains several temporal orderless center components (or units, words, clusters), can encode the variable-length input sequence into a single utterance level vector representation.

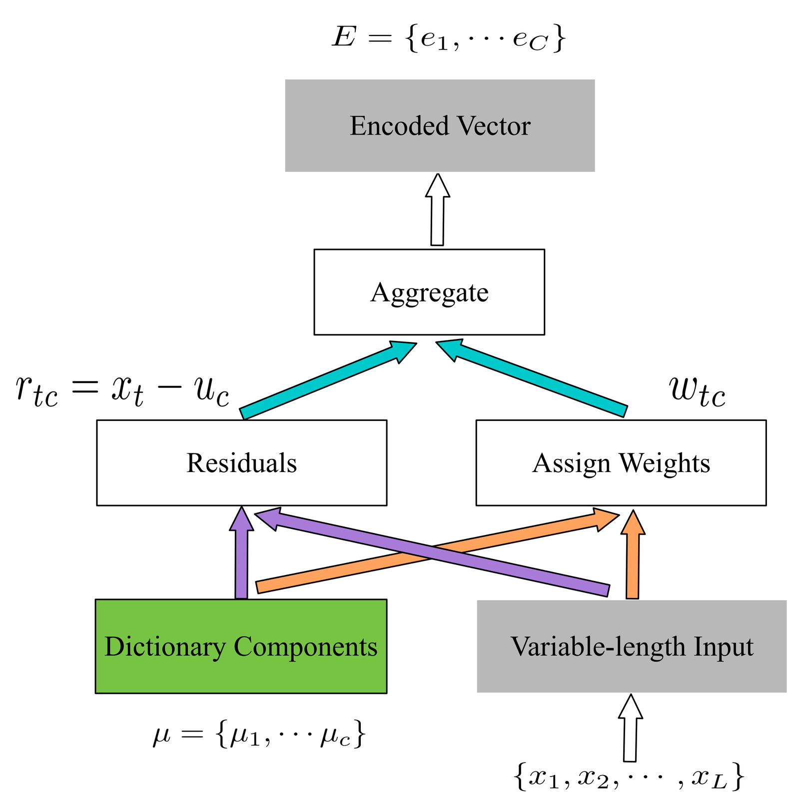
Dictionary Learning

VQ codebook (K-means)
UBM (GMM)
Phoneme decoder (DNN)
Phonotactic tokenizer (GMM / DNN)

Vector Encoding

Average Quantization Distortion
GMM likelihood, GMM Supervector, GMM i-vector
DNN i-vector
Bag-of-words, N-gram token statistics

LDE Implementation



The LDE layer is a directed acyclic graph and all the components are differentiable w.r.t the input $\mathbf{X} = \{\mathbf{x_1}, \mathbf{x_2}, \dots, \mathbf{x_L}\}$ and the learnable parameters. Given a set of L frames feature sequence and a learned dictionary center $\boldsymbol{\mu} = \{\boldsymbol{\mu_1}, \boldsymbol{\mu_2}, \dots, \boldsymbol{\mu_c}\}$, each frame of feature x_t can be assigned with a weight to each component $\boldsymbol{\mu_c}$ and the corresponding residual vector is denoted by

$$\mathbf{r_{tc}} = \mathbf{x_t} - \mathbf{u_c}, \qquad \text{where } t = 1, 2, \dots, L \text{ and c} = 1, 2, \dots, C.$$

The non-negative assigning weight is given by a softmax function,

$$\mathbf{w_{tc}} = \frac{\exp(-\mathbf{s_c}||\mathbf{r_{tc}}||^2)}{\sum_{\mathbf{m=1}}^{\mathbf{C}} \exp(-\mathbf{s_m}||\mathbf{r_{tm}}||^2)}$$

Given the assignments and the residual vector, similar to conventional GMM Supervector, the residual encoding model applies an aggregation operation for every dictionary component center $\boldsymbol{\mu_c}$

$$\mathbf{e_c} = \sum_{\mathbf{t=1}}^{\mathbf{L}} \mathbf{e_{tc}} = \frac{\sum_{\mathbf{t=1}}^{\mathbf{L}} \mathbf{w_{tc}} \times \mathbf{r_{tc}}}{\sum_{\mathbf{t=1}}^{\mathbf{L}} \mathbf{r_{tc}}}$$

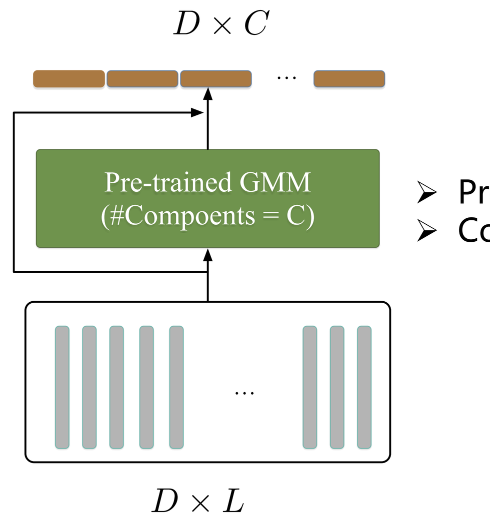
In order to facilitate the derivation we simplified it as

$$\mathbf{e_c} = \sum_{\mathbf{t=1}}^{\mathbf{L}} \mathbf{e_{tc}} = \frac{\sum_{\mathbf{t=1}}^{\mathbf{L}} \mathbf{w_{tc}} \times \mathbf{r_{tc}}}{\mathbf{L}}$$

The LDE layer concatenates the aggregated residual vectors with assigned weights. The resulted encoder outputs a fixed dimensional representation

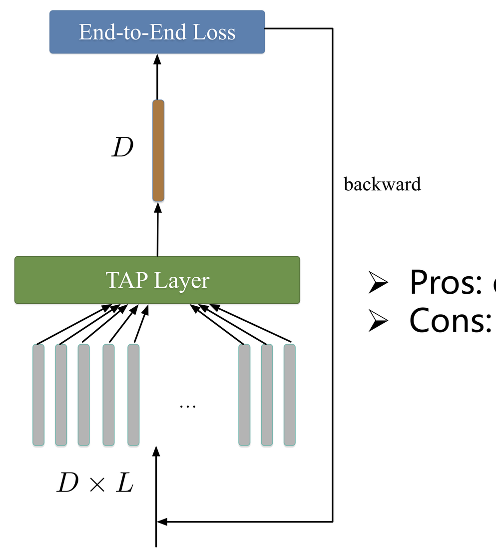
$$\mathbf{E} = \{\mathbf{e_1}, \mathbf{e_2}, \dots, \mathbf{e_C}\}$$

LDE Intuition



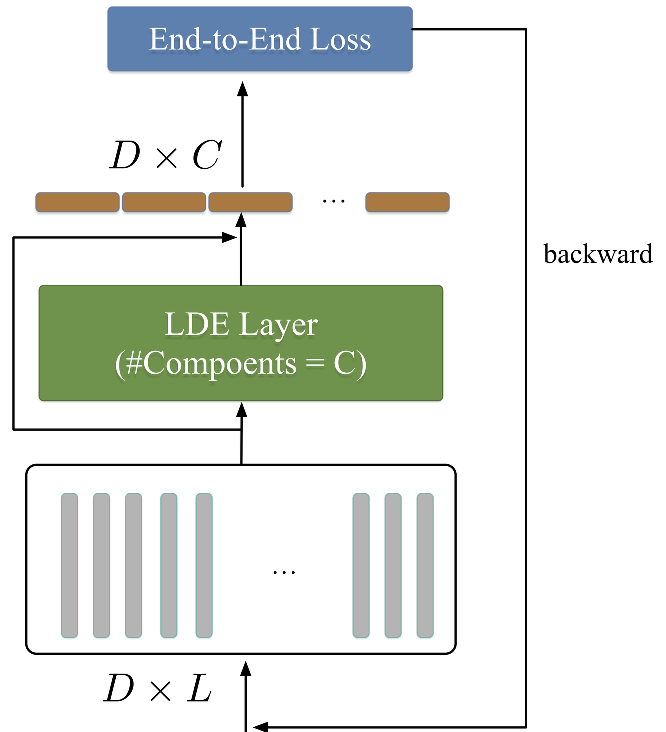
➢ Pros: high-order discriminative statistics
➢ Cons: unsupervised and self-contained

GMM Supervector



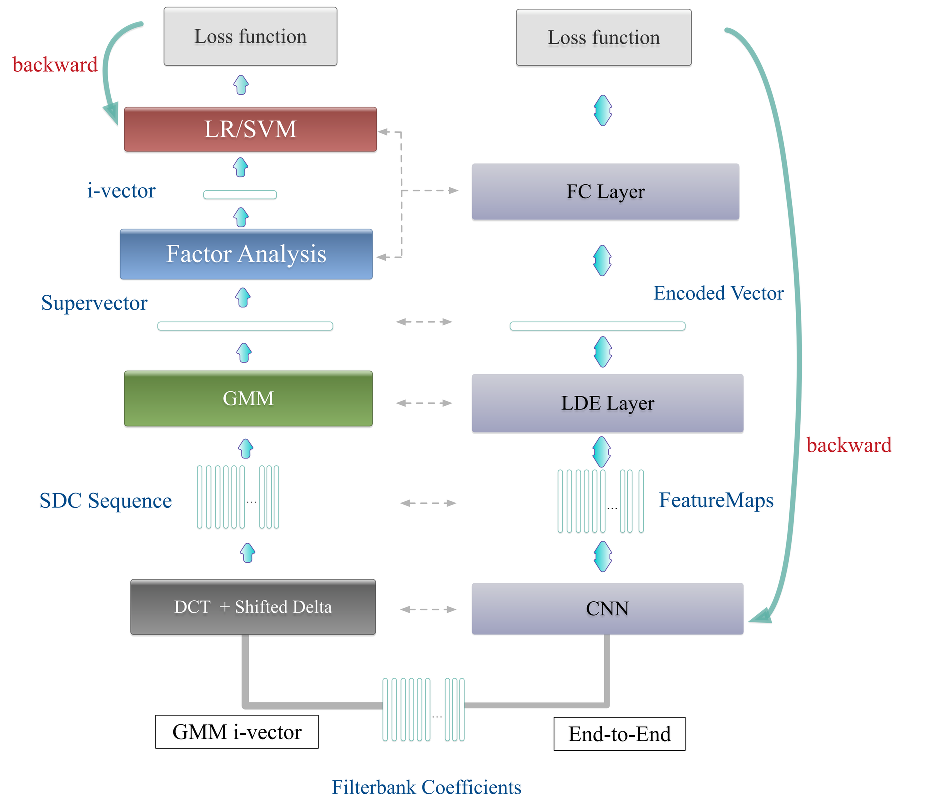
➢ Pros: end-to-end
➢ Cons: simple statistics (mean)

Temporal average pooling



LDE layer

End-to-end neural network with LDE layer



➢ **Motivated by GMM Supervector encoding procedure, we design a learnable dictionary encoding (LDE) layer on top of front-end CNN. The LDE layer simultaneously learns the encoding parameters along with an inherent dictionary in a fully supervised manner.**
➢ **The inherent dictionary is learned from the distribution of the descriptors by passing the gradient through assignment weights. During the training process, the updating of extracted convolutional features can also benefit from the encoding representations.**

Experimental Results and Discussion

layer	output size	downsample	channels	blocks
conv1	$64 \times L_{in}$	False	16	-
res1	$64 \times L_{in}$	False	16	3
res2	$32 \times \frac{L_{in}}{2}$	True	32	4
res3	$16 \times \frac{L_{in}}{4}$	True	64	6
res4	$8 \times \frac{L_{in}}{8}$	True	128	3
avgpool	$1 \times \frac{L_{in}}{8}$	-	128	-
reshape	$128\times L_{out}, L_{out} = \frac{L_{in}}{8}$	-	-	-

- The task of interest is the closed-set language detection. There are totally 14 target languages in testing corpus, which included 7530 utterances split among three nomial durations: 30, 10 and 3 seconds.
- In order to get higher abstract representation better for utterances with long duration, we design a deep CNN based on the well-known ResNet-34 layer architecture, as is described in Table 2. The total parameters of the front-end CNN is about 1.35 million.
- For CNN-TAP system, a simple average pooling layer followed with FC layer is built on top of the font-end CNN. For CNN-LDE system, the average pooling layer is replaced with a LDE layer.
- Because we have no separated validation set, even, we only use the converged model after the last step optimization. For each training step, an integer L within [200,1000] interval is randomly generated, and each data in the mini-batch is cropped or extended to L frames.
- In testing stage, all the 3s, 10s, and 30s duration data is tested on the same model. Because the duration length is arbitrary, we feed the testing speech utterance to the trained neural network one by one.

System ID	System Description	Feature	Encoding Method	$C_{avg}(\%)$			$EER(\%)$		
				3s	10s	30s	3s	10s	30s
1	GMM i-vector	SDC	GMM Supervector	20.46	8.29	3.02	17.71	7.00	2.27
2	CNN-TAP	CNN FeatureMaps	TAP	9.98	3.24	1.73	11.28	5.76	3.96
3	CNN-LDE(C=16)	CNN FeatureMaps	LDE	9.61	3.71	1.74	8.89	2.73	1.13
4	CNN-LDE(C=32)	CNN FeatureMaps	LDE	8.70	2.94	1.41	8.12	2.45	0.98
5	**CNN-LDE(C=64)**	CNN FeatureMaps	LDE	**8.25**	**2.61**	**1.13**	**7.75**	**2.31**	**0.96**
6	CNN-LDE(C=128)	CNN FeatureMaps	LDE	8.56	2.99	1.63	8.20	2.49	1.12
7	CNN-LDE(C=256)	CNN FeatureMaps	LDE	8.77	3.01	1.97	8.59	2.87	1.38
8	Fusion ID2 + ID5	-	-	**6.98**	**2.33**	**0.91**	**6.09**	**2.26**	**0.87**

➢ **The CNN-LDE system outperforms the CNN-TAP system with all different number of dictionary components.**
➢ **When the numbers of dictionary component increased from 16 to 64, the performance improved insistently. However, once dictionary component numbers are larger than 64, the performance decreased perhaps because of overfitting. Comparing with CNN-TAP, the best CNN-LDE-64 system achieves significant performance improvement especially with regard to EER.**
➢ **Besides, their score level fusion result further improves the system performance significantly.**