Convolutional neural network acoustic model for robust Indonesian speech recognition in noisy environment

M J Budiman and D P Lestari
School of Electrical Engineering and Informatics, Bandung Institute of Technology
Bandung, Indonesia
E-mail: marjer17dim@gmail.com

Abstract. Noise causes the decreasing accuracy of automatic speech recognition (ASR). Several techniques have been developed and proposed to overcome this problem. Using artificial neural network (ANN) as acoustic model is one of the techniques. Convolutional neural network (CNN) is a variant of ANN that has been used for acoustic modeling. Another approach is to do pre-processing to the speech signal or to the extracted acoustic feature from speech signal, such as cepstral mean and variance normalization (CMVN). On this work, CNN acoustic models were trained by using CMVN pre-processed acoustic feature to make a noise-robust speech recognition system. Two group of models were made, each to handle 2 kinds of noise (babble noise and street noise). Those acoustic models were tested with noisy speech at different SNR (signal-to-noise ratio) value. Testing results from CNN acoustic models were compared with the ones from Gaussian Mixture Model-Hidden Markov Model (GMM-HMM) acoustic models. Testing results showed the increasing accuracy scores of acoustic models when models were trained using more variation of training data. CNN acoustic models that were trained using FBANK feature have higher accuracy scores than GMM-HMM models that were built using the same feature.

1. Introduction
Noise is some unwanted component in a signal that can cause degradation of automatic speech recognition (ASR) accuracy [1]. Noise causes mismatch between training data and recognition environment. Models which were trained using clean speech have good accuracy score in clean environment, but give poor performance in noisy one. With signal-to-noise ratio (SNR) gets higher, the performance gets worse. Type of noise present in the environment also affect the performance of ASR. Multi-style training is a method that tries to minimize the mismatch, by adding various training data to the training corpus [2]. Another way to improve recognition accuracy is to pre-process acoustic feature, such as applying cepstral mean and variance normalization (CMVN) to acoustic feature.

Deep neural network (DNN) based acoustic models match and even outperform the state-of-the-art performance of Gaussian mixture model (GMM) ASR system. It is known that DNN based models are more robust to the presence of noise [3]. Convolutional neural network (CNN) as one type of DNN has been developed as acoustic models [4] and give better performance in noisy environments [5]. In [5], CNN based acoustic models have 3.9% relative error reduction on average when compared with DNN (feed-forward neural network) based acoustic models.

In this study, we build CNN based acoustic models to improve recognition accuracy in noisy environments. We begin with building speech corpora by augmenting clean speech corpus with 2 types of noise (babble noise and street noise), with each corpus has a certain SNR value. Those
corpora are used to train triphone GMM-HMM acoustic models which act as baseline in this study. Also, by using the corpora and GMM-HMM models, we build CNN based acoustic models. The acoustic features extracted from the corpora are pre-processed by applying CMVN to them. By the end of this study, we obtained a significant decrease in the value of WER from CNN acoustic models, compared to our baseline.

The following sections are organized as follows. Section 2 contains brief explanation about acoustic feature, CMVN, and CNN in general, and CNN as acoustic model. The next section explains more about our CNN acoustic model experiments in this research. The experiments are described step by step, from speech corpus acquisition, corpus augmentation, until the training of CNN acoustic models. The final section discussed about result and analysis based on the experiment results.

2. Feature Extraction and CMVN

2.1. Feature Extraction

Feature extraction is one of few components that form ASR. In feature extraction, speech signal is processed by some operations to obtain acoustic features. The acoustic features then used in model training and decoding process. There are many kinds of acoustic feature used in acoustic modeling, such as MFCC, PLP, FBANK, etc.

In GMM-HMM based acoustic modeling, mel-frequency cepstral coefficient (MFCC) is the most commonly used acoustic feature in GMM-HMM. MFCC processing [6] is shown in Figure 1. Each frame in speech signal produce 12 MFCCs, followed by 1 energy coefficient, resulting with 13 coefficients. Those 13 coefficients often accompanied with 13 delta coefficients and 13 delta-delta coefficients. MFCC is chosen because of uncorrelated characteristics in each of its coefficients. Uncorrelated nature of those coefficients reduces the number of GMM parameters that need to be learned when training acoustic model [6].

![Figure 1. Steps to obtain MFCC feature](image1)

Another type of acoustic feature is filterbank (FBANK) feature. Producing filterbank feature from speech signal is done similar to producing MFCC, without performing discrete cosine transform (DCT) [4]. FBANK feature is often called mel-frequency spectral coefficient (MFSC). Another difference is the number of mel-scale filters used in producing FBANK; it is common to use 40, unlike MFCC which usually uses 23 filters. Without DCT, the resulting FBANK coefficients are highly correlated, and thus are used in DNN based acoustic modeling, because DNN can take advantage of highly correlated inputs. The scheme to produce FBANK is shown in Figure 2.

![Figure 2. Steps to obtain FBANK feature](image2)
2.2. CMVN
Both types of feature can be pre-processed, by applying CMVN. Steps of CMVN begin with subtracting mean of all coefficients from all coefficients. Then, the each of the resulting coefficients is divided with the standard deviation (square root of variance) of all subtracted coefficients. Formula (1) shows the coefficients subtraction and formula (2) shows the division of the coefficients,

\[ X[k] = C[k] - \mu_C \] (1)
\[ Y[k] = \frac{X[k]}{\sigma_X} \] (2)

with \( C \) is the initial coefficients, \( \mu_C \) is the mean of coefficients, and \( \sigma_X \) is the standard deviation of the subtracted coefficients.

3. Convolutional Neural Network
Convolutional neural network (CNN) is a variant of neural network, that is inspired from how human vision system works [7]. It is firstly used in image classification task. CNN can take inputs in the form of vectors (1 dimensional), matrices (2 dimensional), and higher dimensional data structures. The difference between CNN and plain neural network is the emergence of convolution layer and pooling layer. Usually a CNN contains some convolution layers, some pooling layers, and some fully connected layers.

In convolution layer, there are data structures that are called filters/kernels. Each kernel can take the form of a vector, matrix, or data structure with higher dimension. These filters and the inputs of convolution layer are used as operands to convolution operation. Each element in the inputs is multiplied with the respecting element in the kernels, then the results is summed. Many kernels can be used for one single input. Convolution layer makes the dimension of output is smaller. The illustration of how convolution layer works is shown in Figure 3. On Figure 3, the kernel is a vector that has 3 elements, those are \(<0, 1, 0>\), so the kernel size is 3. After doing convolution operation of the first region of input, the kernel is moved along a certain axis for 1 unit. The number of units is called stride.

![Figure 3. Illustration of convolution layer](image)

In pooling layer, the dimension of output is smaller. The illustration of how pooling layer works is shown in Figure 4. On Figure 4, the kernel is a vector that has 1 element, those are \(max\). After doing pooling operation of the first region of input, the kernel is moved along a certain axis for 1 unit. The number of units is called stride.

![Figure 4. Illustration of pooling layer](image)
The next layer of CNN is the pooling layer. In this layer, reduction of input dimension is done again, using some aggregate function, over some region of the input. Common functions used in this layer is max (max-pooling) and average (average-pooling). Similar to convolution layer, there is kernel size (instead in this layer it’s called pool size) and a certain stride value. Figure 4 shows the illustration of a max-pooling layer, with pool size 3 and stride 1.

The final layers of CNN are fully connected layers. Inputs to the fully connected layers must be in the form of a vector. The all outputs from the previous layer is flattened to form a single vector. A particular activation function can be applied to the output of convolution, pooling, or fully-connected layer. The simple topology of CNN is shown in Figure 5.

Acoustic model in [4] used CNN with one convolution layer, one pooling layer, and 2 fully connected layers. The details of each layer are described in Table 1. The complete topology is shown in Figure 6.

| Layer     | Description                        |
|-----------|------------------------------------|
| Convolution | • Number of filters: 150            |
|           | • Filter size: 8                   |
|           | • Stride: 2                        |
| Pooling   | • Pool size: 6                     |
|           | • Stride: 2                        |
| Hidden    | • Number of neurons: 1024          |
|           | • Activation function: ReLU        |
| Output    | • Number of neurons: number of states in HMM |
|           | • Activation function: softmax     |

Training of a DNN or CNN based acoustic models, requires that each feature vector (input) has a certain label, that is the HMM state id associated with that frame. To assign labels on all frames of input, alignment must be done using a previously trained GMM-HMM acoustic model. So, to build a CNN based acoustic model, a GMM-HMM acoustic model must be made first.

4. Experiment

4.1. Data Acquisition and Augmentation
Feature extraction is one of few components that form ASR. In feature extraction, speech signal is processed by some operations to obtain acoustic features. The acoustic features then used in model training and decoding process. There are many kinds of acoustic feature used in acoustic modeling, such as MFCC, PLP, FBANK, etc.

![Topology of CNN for speech](image)

**Figure 6.** Topology of CNN for speech

Testing corpus, obtained from PT. Prosa Solusi Cerdas is also a clean speech corpus, recordings of dictated speech from 30 people, each of them uttered 20 Indonesian sentences. Total duration of the corpus is about 57 minutes. The difference is each sentence they read is only comprised of digits. The digits contained in each sentence are limited to 0 until 9. The reason why testing corpus only contains reading of digits, is to maximize the accuracy of ASR. Lesser the number of vocabularies in the lexicon makes accuracy of ASR gets higher. Focus of this research is to measure accuracy of ASR in noisy environment, so any other factor that degrades accuracy should be minimized. Transcripts and lexicon are also included in this corpus.

Aside from clean speech, recordings of noise were also gathered. These recordings were used to augment the clean speech corpus, to yield new corpus with a certain noise. There are 2 noise recordings gathered, babble noise and street noise. Each of them downloaded from www.orangefreesounds.com and has duration of 90 seconds.

Data augmentation is done with a tool called Sox. Sox can do some manipulation to audio files, such as mixing them, trimming, adjusting volumes, etc. For each utterance in clean speech corpus, its duration is calculated. Then a segment with having such duration is sliced from noise recording. The segment volume is adjusted to obtain a certain SNR value, then it is mixed with clean speech utterance.
Two groups of corpora are yielded from the augmentation process, and are shown in Table 2 and Table 3. First group is corpora that contain babble noise (identified with “BB” infix) and second group is corpora that contain street noise (identified with “SR” infix). SNR value of each corpus is shown in the suffix of the corpus name. Suffix “20_10_00” means a corpus made from combination of SNR 20 dB, 10 dB, and 0 dB corpora.

### Table 2. Corpora for training

| No. | Corpus Name     | No. | Corpus Name     |
|-----|-----------------|-----|-----------------|
| 1   | TR_CLEAN        | 5   | TR_SR_20       |
| 2   | TR_BB_20        | 6   | TR_SR_20_10    |
| 3   | TR_BB_20_10     | 7   | TR_SR_20_10_00 |
| 4   | TR_BB_20_10_00  |     |                |

### Table 3. Corpora for testing

| No. | Corpus Name     | No. | Corpus Name     |
|-----|-----------------|-----|-----------------|
| 1   | TS_CLEAN        | 7   | TS_SR_20       |
| 2   | TS_BB_20        | 8   | TS_SR_15       |
| 3   | TS_BB_15        | 9   | TS_SR_10       |
| 4   | TS_BB_10        | 10  | TS_SR_05       |
| 5   | TS_BB_05        | 11  | TS_SR_00       |
| 6   | TS_BB_00        |     |                |

**4.2. Baseline**

Baseline for this experiment is obtained from a collection of triphone GMM-HMM acoustic models. Those models are built using the training corpuses, each corpus becomes one model. FBANK acoustic feature is used to build these models. CMVN is not applied for the acoustic feature. Kaldi speech recognition toolkit is used to do feature extraction, CMVN, and GMM-HMM modeling. The models are used to recognize (decode) audio files from testing corpuses, then the WER is computed by comparing recognition result and transcripts of testing corpuses. Decoding is also done by Kaldi. Table 4 shows average WER for each SNR value of testing corpus, and Table 5 shows average WER for each training corpus.

### Table 4. Average WER on each SNR value (baseline)

| Models          | SNR Values |
|-----------------|------------|
|                 | Clean 20 dB| 15 dB | 10 dB | 5 dB | 0 dB |
| Babble Noise    | 18.91 24.60| 29.16 | 41.70 | 63.98 | 85.05 |
| Street Noise    | 25.88 27.70| 38.88 | 63.43 | 86.45 | 92.67 |
Table 5. Average WER on each training corpus (baseline)

| Models         | Clean | 20 dB | 20, 10 dB | 20, 10, 0 dB |
|----------------|-------|-------|-----------|-------------|
| Babble Noise   | 72.68 | 45.36 | 29.98     | 27.58       |
| Street Noise   | 79.33 | 55.11 | 45.35     | 43.54       |

4.3. CNN Modeling

CNN acoustic models in this research are built using FBANK acoustic feature. CMVN was applied on the extracted feature vectors. GMM-HMM models from baseline is used for the alignment process. Alignment is done using Kaldi. The training scheme of CNN acoustic models is shown in Figure 7.

![Figure 7. Training scheme of CNN acoustic models](https://example.com/fig7)

CNN is built using topology from [4]. Keras library is used to compose the CNN, layer by layer. To connect CNN built with Keras and alignment process by Kaldi, an interface from [https://github.com/dspavankumar/keras-kaldi](https://github.com/dspavankumar/keras-kaldi) is used. In the beginning the interface was made to build DNN based acoustic models, so it was modified to accommodate CNN acoustic models.

CNN training is using holdout validation schema, with 20% data is used as validation set, and 80% data as training set. The training is using hyperparameter as described in Table 6.

CNN models are used to decode the same testing corpus, then WER scores are also measured. The result is shown in Table 7 (comparison for each SNR value) and Table 8 (comparison for each training corpus).

Results from Table 4 and Table 7 show that CNN acoustic models, when tested with 0 dB SNR corpus give 20.89% (babble noise model) and 3.9% (street noise model) error reduction compared to respective GMM-HMM models. In other SNR value, the error reduction ranges between 48.2 – 88.4% (babble noise model) and 7.8 – 88.1% (street noise model). These results thus prove that CNN acoustic models built with CMVN pre-processed features have better accuracy in noisy environment. The results from Table 5 and Table 8 show that by using more training data and more variation of SNR value in training data, the WER gets lower. CNN models trained with the largest corpus give
17.8% (babble noise model) and 14.4% (street noise model) error reduction. From results in all tables, it is obvious that babble noise models have lower error rate than street noise models.

**Table 6.** Details of CNN hyperparameters

| Hyperparameter   | Value                      |
|------------------|----------------------------|
| Learning rate    | 0.1                        |
| Batch size       | 512                        |
| Momentum         | 0.5                        |
| Loss function    | Sparse categorical cross-entropy |

**Table 7.** Average WER on each SNR value (CNN)

| SNR Values | Models        | Clean | 20 dB | 15 dB | 10 dB | 5 dB | 0 dB |
|------------|---------------|-------|-------|-------|-------|------|------|
|            | Babble Noise  | 2.20  | 2.82  | 5.46  | 13.05 | 33.12| 67.28|
|            | Street Noise  | 3.07  | 13.34 | 28.51 | 53.30 | 79.68| 89.02|

**Table 8.** Average WER on each training corpus (CNN)

| Training Corpora | Models     | Clean | 20 dB | 20, 10 dB | 20, 10, 0 dB |
|------------------|------------|-------|-------|-----------|-------------|
| Babble Noise     | 23.24      | 20.46 | 19.82 | 19.10     |             |
| Street Noise     | 48.47      | 46.78 | 41.22 | 41.48     |             |

5. **Conclusion**

In this research we build CNN acoustic models to build robust Indonesian acoustic models in noisy environment. By using CMVN and multi-style training, CNN models we build give lower WER on all SNR value of testing data, compared to GMM-HMM acoustic models. On the noisiest test data (0 dB), we get 20.89% error reduction from babble noise models and 3.9% error reduction from street noise models.

Based on our experiment results, we encourage the future research on noise-robust Indonesian ASR to develop large vocabulary ASR with robust-to-noise characteristics. We suggest to add more data to the training set and use testing set that contains large vocabulary. We also suggest to test acoustic models in real environment or using recordings from real environment, not the ones from augmentation process.

**Acknowledgements**

We thank PT. Prosa Solusi Cerdas and other parties who contributed in this experiment or during the process of working on this paper. We thank everyone who gives their time to record their voice.
References
[1] Benesty J, Chen J, Cohen I and Huang Y 2009 Noise Reduction in Speech Processing (New York: Springer)
[2] Li J, Deng L, Gong Y and Haeb-Umbach R 2014 Overview of noise-robust automatic speech recognition IEEE/ACM Transactions on Audio, Speech, and Language Processing 22 pp 745–77
[3] Li J, Deng L, Gong Y and Haeb-Umbach R 2016 Robust Automatic Speech Recognition A Bridge to Practical Application (Cambridge: Academic Press)
[4] Abdel-Hamid O, Mohamed A, Jiang H, Deng L, Penn G and Yu D 2014 Convolutional neural networks for speech recognition IEEE/ACM Transactions on Audio, Speech, and Language Processing 22 pp 1533–45
[5] Huang J, Li J and Gong Y 2015 An analysis of convolutional neural networks for speech recognition 2015 IEEE Int. Conf. on Acoustics, Speech, and Signal Processing pp 4989–93
[6] Jurafsky D and Martin J 2008 Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition (New Jersey: Prentice Hall)
[7] Goodfellow I, Bengio Y and Courville A 2017 Deep Learning (Cambridge: MIT Press)
[8] Lestari D P, Iwano K and Furui S 2006 A large vocabulary continuous speech recognition system for Indonesian language 15th Indonesian Scientific Conf. in Japan pp 17–22