딥 러닝을 이용한 음성인식 오류 판별 방법

김현호○, 윤승, 김상훈
과학기술연합대학원대학교○, 한국전자통신연구원
hyunho.kim@etri.re.kr, syun@etri.re.kr, ksh@etri.re.kr

Speech Recognition Error Detection Using Deep Learning

Hyun-Ho Kim○, Seung Yun, Sang-Hun Kim
Korea University of Science and Tehcnology○,
Electronics and Telecommunications Research Institute

요 약

자동통역(Speech-to-speech translation)의 최우선 단계인 음성인식과정에서 발생한 오류문장은 대부분 비문법적 구조를 갖거나 의미를 이해할 수 없는 문장들이다. 이러한 문장으로 자동번역을 할 경우 심각한 통역오류가 발생하게 되어 이에 대한 개선이 반드시 필요한 상황이다. 이에 본 논문에서는 음성인식 오류문장이 정상적인 인식문장에 비해 비문법적이거나 무의미하다는 특징을 이용하여 DNN(Deep Neural Network) 기반 음성인식오류 판별기를 구현하였으며 84.20%의 오류문장 분류성능결과를 얻었다.

주제어: Deep Learning, 음성인식, Classification, Word embedding

1. 서론

음성인식 오류는 여러 원인을 가지고 있지만 잘못된 음성인식 결과 중에서 대부분은 표1에 나열된 예시문장들처럼 비문법적이거나 의미를 이해하기 어려운 문장들이 포함되어 있다. 이러한 특성을 이용해 문장을 구별해 낸다면, 음성인식의 오인식된 문장 다수를 자동으로 구별해낼 수 있게 된다.

표1 음성인식 오류문장의 예

정인식 결과	엄마랑 애기랑 함께 있어요.
오인식 결과	흐윽 리플 해 인아 너흰이랑 함께 입금 아니오.
정인식 결과	뭐라구. 175센티미터.
오인식 결과	네 뭘로 볼 만한 데는 별 성과가 175 센티미터 보구.
정인식 결과	사랑해요. 사랑해요.
오인식 결과	인터넷 사랑이 누구 사랑의 이유.

최근에 뛰어난 분류성능을 보여주고 있는 DNN(Deep Neural Network)은 음성분야와 이미지 처리영역뿐만 아니라 문서나 문장의 주제분류 문제에서도 높은 성능을 보여주었다[1].

본 논문에서는 오인식된 문장과 정상적으로 인식된 문장을 학습한 DNN으로 음성 인식의 오류 여부를 판별하는 분류기를 구축하였다.

음성인식 오류 판별기를 만들어낸다면, 음성인식 오류가 발생한 경우에 자동으로 사용자에게 재발화하도록 요청하는 시스템을 구축할 수 있고, 오류 탐지결과에 대한 사용자의 의견을 학습데이터에 반영하거나, 음성인식 성능을 측정하는 평가데이터로 사용한다면 점진적인 음성인식 성능 개선에도 활용할 수 있다.

2. 관련 연구

DNN은 이미지 분류 문제에서 가장 높은 정확도를 가지고 있다[2]. 또한, 대용량 어휘 음성인식(LVCSR) 분야에서는 DNN과 HMM(Hidden Markov Model)을 결합한 음향모델이 기존에 표준으로 사용되었던 GMM(Gaussian Mixture Mode) 방식보다 훨씬 정확한 음성인식 성능을 보여주었다[3]. 그뿐만 아니라 많은 자연어처리 문제에도 적용되어 높은 성능을 보여주었다. 예를 들어, 형태소 분석, 청킹, 개체명 인식, 의미역 결정[4], 문서 주제 분류[1], 문장분류[5], 감정분석[6] 분야가 있다.

문서 분류문제는 문서에 포함된 단어의 영향력이 크기 때문에 간단히 bag-of-word만 활용하더라도 높은 분류 성능을 보이지만 본 논문이 해결하려는 문제는 단어뿐만 아니라 문법구조와 의미까지 특징으로 추출되어야 하기 때문에 텍스트 주제 분류 문제[1]보다 더 까다로운 문제로 여겨진다.

3. 실험 설계

3.1 DNN 구조

DNN은 제한된 볼츠만 모델을 여러 층으로 쌓은 구조를 가지고 있다. 각 층은 일렬의 노드로 구성되어 있으며, 각 층간에는 weighted edge로 연결되어 있다. 그리고 DNN의 층을 이루는 노드들은 각각 Activation 함수를 가지고 있다. Activation 함수로 사용되는 고전적인 예는 sigmoid 함수 또는 tanh 함수 등이 있으나, 본 실험에서는 비선형적 특성을 가진 Rectified Linear Unit을

Activation 함수로 선택하였다.

ReLU(Rectified Linear Unit)는 입력 값이 0 이상인 경우에 활성화되고 0보다 작은 값이 입력되는 경우, 비활성화되어 0을 출력하는 원리로 동작한다. ReLU가 활성화된 경우에는 미분 값이 1이기 때문에, 인공신경망 내부의 히든 노드들 사이에 전달되는 경사도 값(gradient)이 사라지는 gradient vanishing 문제가 존재하지 않는 효과를 가지고 있다. 그리고 ReLU 함수는 활성화 되지 않은 경우에 0의 값을 갖기 때문에 활성화 여부가 분명하며, 이러한 특징은 분류문제에 큰 역할을 한다[3].

$$h^{(i)} = \max(w^{(i)T}x, 0) = \begin{cases} w^{(i)T}x & w^{(i)T}x > 0 \\ 0 & \text{else} \end{cases}$$

그림1 Rectified Linear Unit[3]

출력 층(Output Layer)은 정상문장인지 오류문장인지 판별 결과가 출력되는 2개의 노드로 구성하고 그 결과가 softmax 함수를 거치면서 최종 분류결과를 얻도록 하였다.

DNN 분류기를 구현하는 과정에서는 GPU 연산에 최적화된 다차원 행렬연산 라이브러리인 Theano[7]를 사용하였다.

3.2 데이터 구축

음성인식 오류 판별기를 구축하기 위하여, 학습데이터는 ETRI 음성인식기의 인식결과 데이터를 사용하였으며, 수작업으로 오류문장과 정상문장으로 구분하여 구축하였다. 그리고 초기 실험에서는 8만 문장을 사용하여 학습하였고, 추후에 36만 문장으로 확장하여 실험 결과를 비교하였다.

평가 데이터는 정상문장과 오류문장 각각 4000문장으로 구성하였고, 과적합(Overfitting)여부를 판단하기 위해 학습데이터 내에 존재하는 평가데이터와 학습데이터에 존재하지 않는 평가데이터로 구성하여 평가를 진행하였다.

그리고 동형이의어의 모호성을 피하기 위한 목적으로 형태소 분석기[8]를 사용하여 단어에 형태소 정보를 반영하였다.

학습에 사용된 문장의 길이는 부록의 그림2에서 볼 수 있듯이 대부분 20어절이하로 이루어져 있고 20어절보다 긴 경우는 전체 data에서 매우 극히 작은 비중을 차지하고 있었다. 그러한 이유로 문장의 길이를 최대 20어절로 제한하여 실험을 진행하였다.

DNN에 문장을 입력하기 위해서는 문장을 벡터로 표현해야 한다. 대용량의 비구조적 텍스트 데이터를 벡터로 표현하기 위해서 Word2Vec 방법[9][10]을 선택하였고, Python으로 구현된 Gensim 라이브러리[11]에 내장된 모듈을 사용하였다.

4. 실험 결과

4.1 DNN 음성인식 오류 판별기 성능 측정

이 장에서는 DNN 음성인식 오류문장 판별기의 평가결과를 확인하였다. 초기 실험은 학습데이터 8만개를 사용하였고, 실험 결과는 부록의 그림3에서 볼 수 있다. 학습 데이터에서 추출한 데이터로 평가한 결과는 99.80%의 정확도를 얻었지만, 학습데이터에 존재하지 않은 데이터로 평가한 결과는 74.64%로서 두 평가결과 간에 큰 차이를 보여주었다. 즉, 과적합문제로 인해 낮은 분류성능을 확인할 수 있었다.

4.2 Overfitting 문제 해결

과적합문제를 해결하기 위한 효과적인 방법으로는 선행학습(Pretraining)방법과 Dropout이 있다[12]. 두 방법 중에서 Dropout을 도입하여 과적합문제를 개선하였다. 그리고 과적합을 줄이기 위한 가장 일반적이고 쉬운 방법은 학습데이터의 양을 인위적으로 늘리기이다[2]. 그렇기 때문에 학습데이터를 증강하여 추가적으로 성능을 향상 시켰다.

4.2.1 Dropout기술 도입

부록의 그림4는 DNN에 Dropout을 적용한 후에 오류 문장 분류 성능 평가 결과의 변화추이를 보여주고 있다. 이전 실험에서는 과적합의 문제가 확연히 드러났으나, Dropout을 적용 후에는 분류성능이 최대 82.42%까지 정확도가 높아져서, ERR(Error Reduction Rate)이 29.96% 향상되었다.

4.2.2 데이터 증강

Dropout 도입에 더하여, 학습데이터를 4.5배 늘려서 실험을 진행하였으며, 부록의 그림5와 같이 오류 문장 분류 성능 평가 결과가 최대 84.20%로 향상되었다. 이는 초기 실험대비 ERR이 37.69% 개선되는 큰 성과이다.

4.3 DNN 최종 출력 값 편차를 이용한 분류성능 개선

Dropout의 도입과 데이터 증강을 통해서 정확도를 크게 개선했으나, 실제 음성인식기에 반영하기 위해서는 더 높은 정확도가 필요하다. 그래서 본 논문에서는 DNN의 최종 출력 값 사이의 차가 클수록 더 높은 신뢰도를 가질 것이라 판단되어, 출력된 두 값의 차이에 따른 분류 성능을 확인해보았다.

부록의 그림6에서처럼 Dropout을 적용하지 않고, 8만 문장으로 학습한 DNN은 평가셋 전체에 대해서 분류성능이 74.64%였으며 편차가 가장 큰 경우에는 75.58%로 평균과 큰 차이를 보이지 못했기 때문에, 출력 값의 편차에 따라 정확도의 변화는 없었다. 반면에 데이터를 증강하고 Dropout을 적용한 후에는 출력 값의 편차가 가장 큰 경우 97.16%까지 정확도가 높아졌다. 그리고 편차가 큰 상위 30%의 평가 데이터에 대해서는 95.00%의 정확도를 얻을 수 있었다.

5. 실험 평가 및 분석

음성인식 오류 판별이라는 목적으로 DNN을 이용한 기본방법론이 만들어졌으며, 음성인식 오류 판별에 대해 유의미한 실험 결과를 보임을 확인할 수 있었다. 그러나 음성인식 시스템에 실제로 적용하기 위해서 데이터의 확장이 필요한 반면, 훈련데이터로 사용되는 음성인식오류 문장 확보에 어려움이 있다. 그렇기 때문에 훈련데이터 확장 방안을 모색할 필요가 있다. 그리고 오류 판별기의 출력 값 편차가 큰 데이터에 대하여 상위 30%기준으로 95.00%의 신뢰도를 보였지만 이 기준을 결정하는 규칙에 대한 연구 또한 필요하다.

6. 결론 및 추후 연구

본 논문에서는 DNN을 활용하여 음성인식 결과에 오류가 있는지 여부를 판별하는 이진분류기를 구축하였다. 컴퓨터비전 또는 음성인식 연구분야에서와 마찬가지로 인공신경망의 과적합문제가 발생하였으며 Dropout을 도입하고, 데이터를 증강하여 분류성능을 크게 높였다. 본 실험에서 설계한 DNN 모델은 의미를 알 수 없는 비정상적인 문장이 음성인식 결과로 출력되는 경우에 대한 해결책으로 활용할 수 있지만 유사한 발음으로 인해 발생한 음성인식 오류문제는 해결할 수 없다. 그렇기 때문에 향후에는 유사발음으로 인해 발생된 음성인식 오류에 대한 방안을 모색하고자 한다.

사사

본 연구는 한국전자통신연구원 연구운영비지원사업의 일환으로 수행하였음. ["언어장벽없는 국가구현을 위한 자동통번역산업 경쟁력 강화사업 (15ZS1110)"]

참고문헌

[1] ZHANG, Xiang; LECUN, Yann. Text Understanding from Scratch. arXiv preprint arXiv:1502.01710. 2015.

[2] KRIZHEVSKY, Alex; SUTSKEVER, Ilya; HINTON, Geoffrey E. Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems. pp. 1097-1105, 2012.

[3] MAAS, Andrew L.; HANNUN, Awni Y.; NG, Andrew Y. Rectifier nonlinearities improve neural network acoustic models. In: Proc. ICML. 2013.

[4] COLLOBERT, Ronan, et al. Natural language processing (almost) from scratch. The Journal of Machine Learning Research. 12: pp.2493-2537, 2011.

[5] KIM, Yoon. Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882, 2014.

[6] ZHOU, Shusen; CHEN, Qingcai; WANG, Xiaolong. Active deep networks for semi-supervised sentiment classification. In: Proceedings of the 23rd International Conference on Computational Linguistics: Posters. Association for Computational Linguistics. pp.1515-1523, 2010.

[7] BERGSTRA, James, et al. Theano: A CPU and GPU math compiler in Python. In: Proc. 9th Python in Science Conf. pp.1-7, 2010.

[8] Eunjeong L. Park, Sungzoon Cho. “KoNLPy: Korean natural language processing in Python”, Proceedings of the 26th Annual Conference on Human & Cognitive Language Technology, Chuncheon, Korea. pp.133-136, 2014.

[9] MIKOLOV, Tomas, et al. Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems. pp.3111-3119, 2013.

[10] MIKOLOV, Tomas, et al. Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781, 2013.

[11] ŘEHŮŘEK, Radim; SOJKA, Petr. Software framework for topic modelling with large corpora. 2010.

[12] 이창기, 김준석, 김정희. 딥 러닝을 이용한 한국어 의존 구문 분석. 한글 및 한국어 정보처리 학술대회. pp.87-91, 2014.

부록

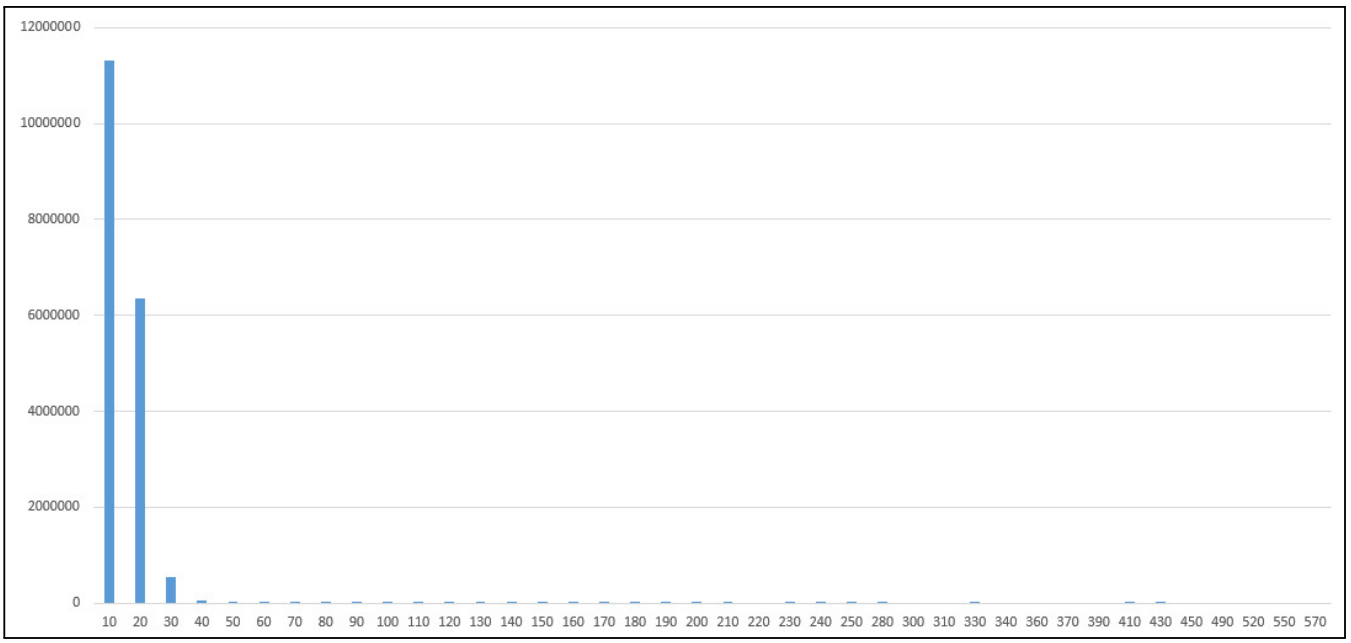


그림2 문장 길이 별 분포 그래프

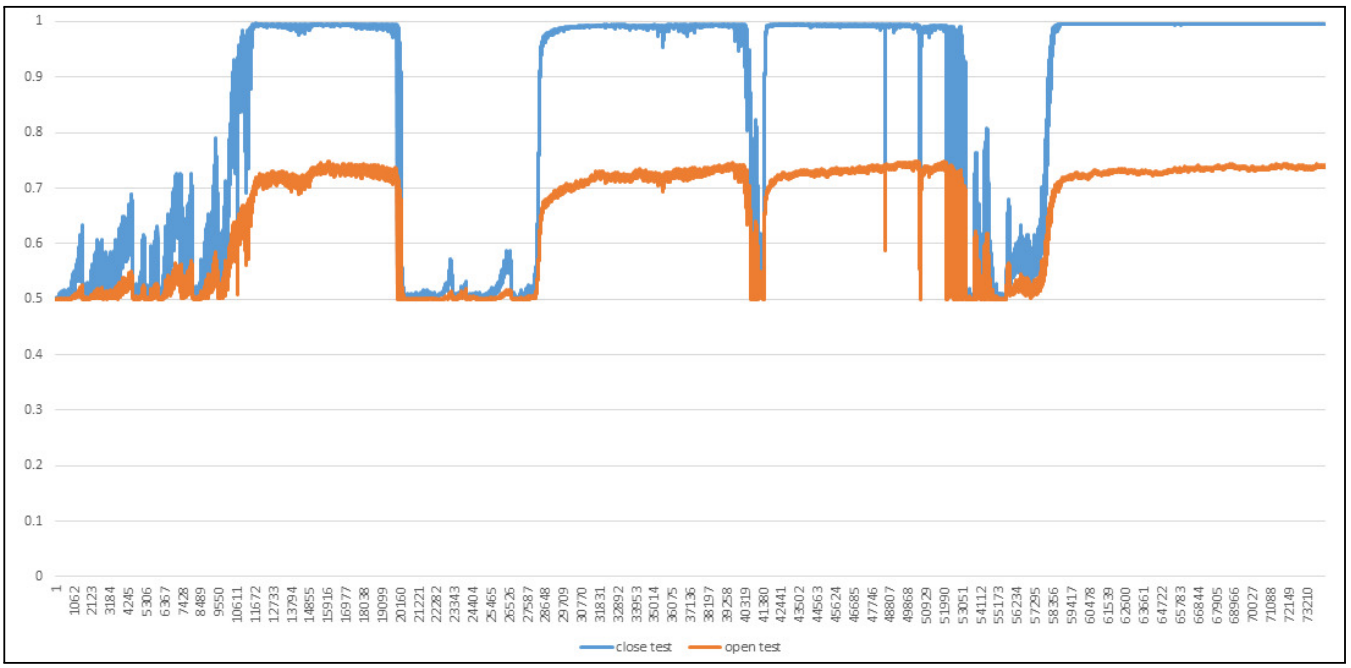


그림3 평가 정확도 변화 그래프

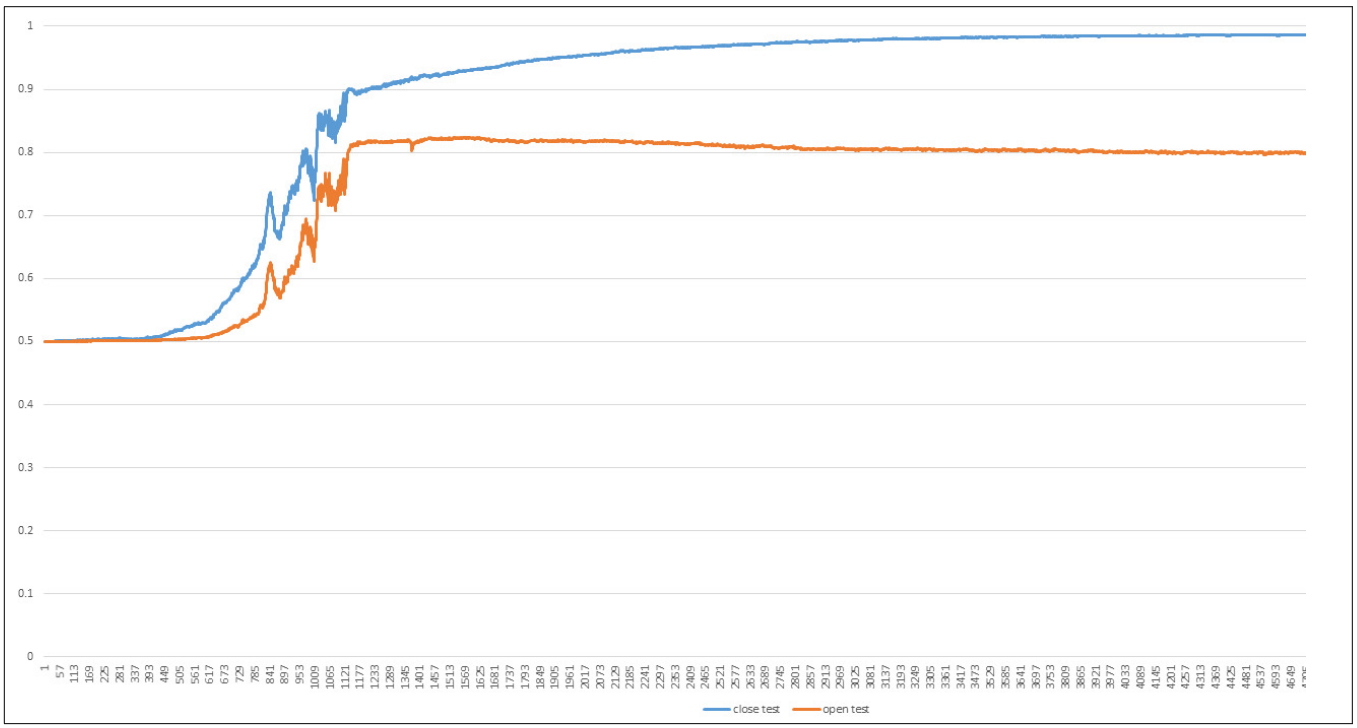


그림4 Dropout 도입 후 평가 정확도 변화 그래프

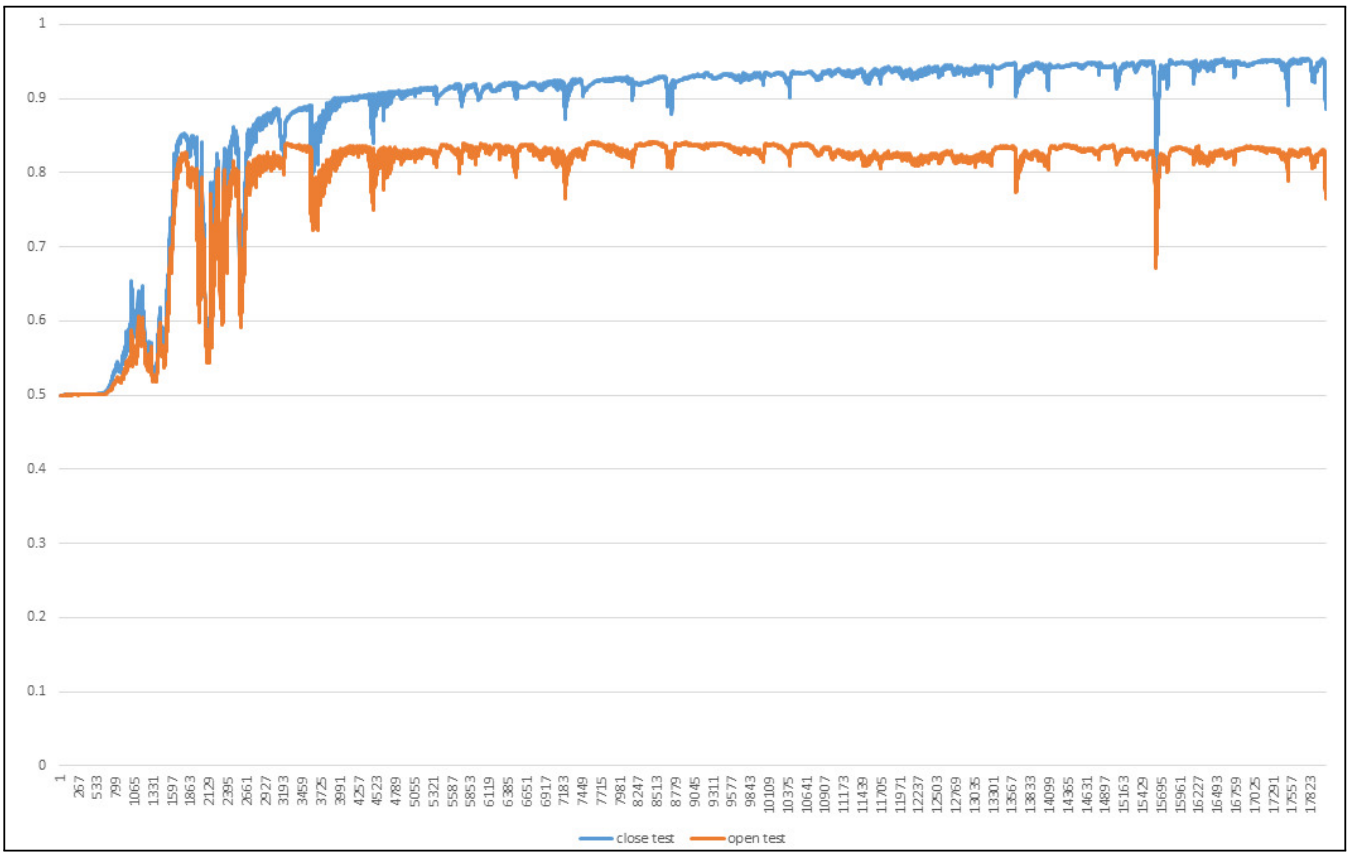
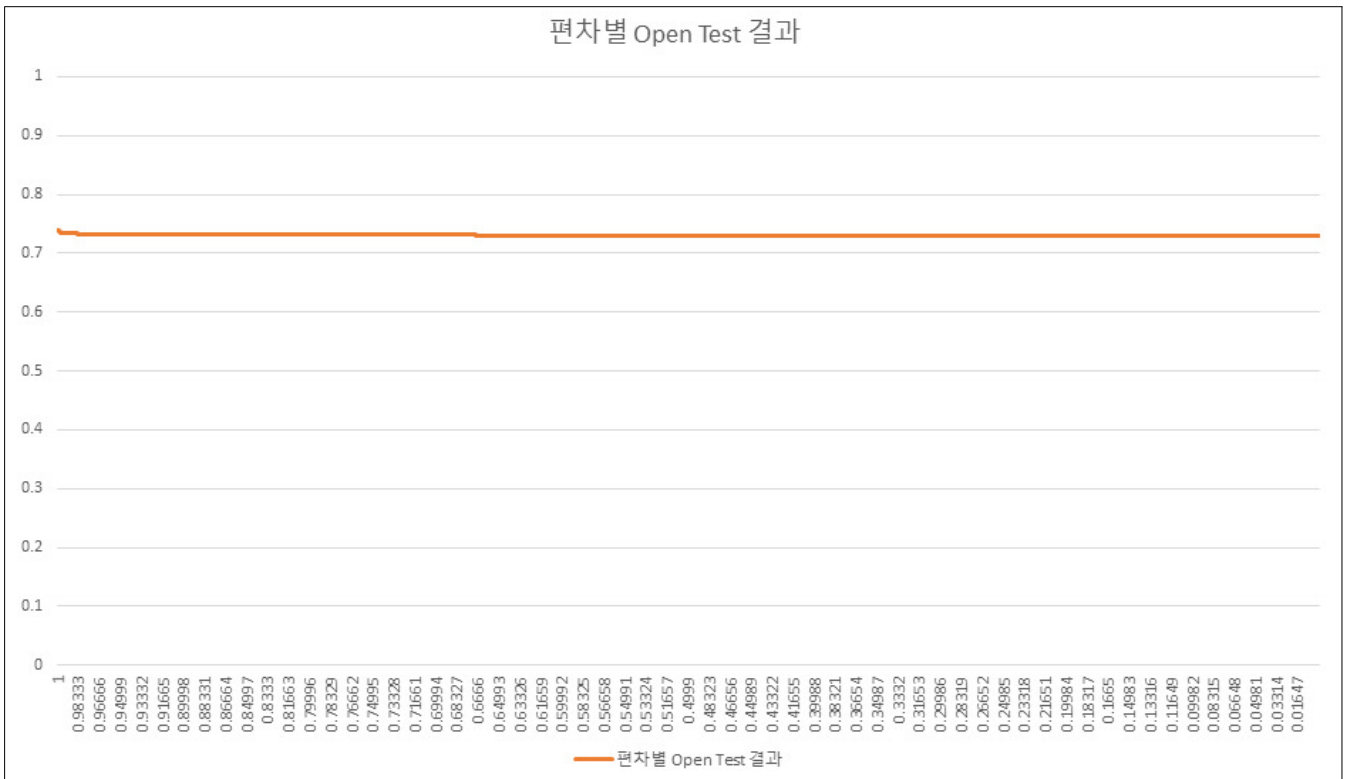
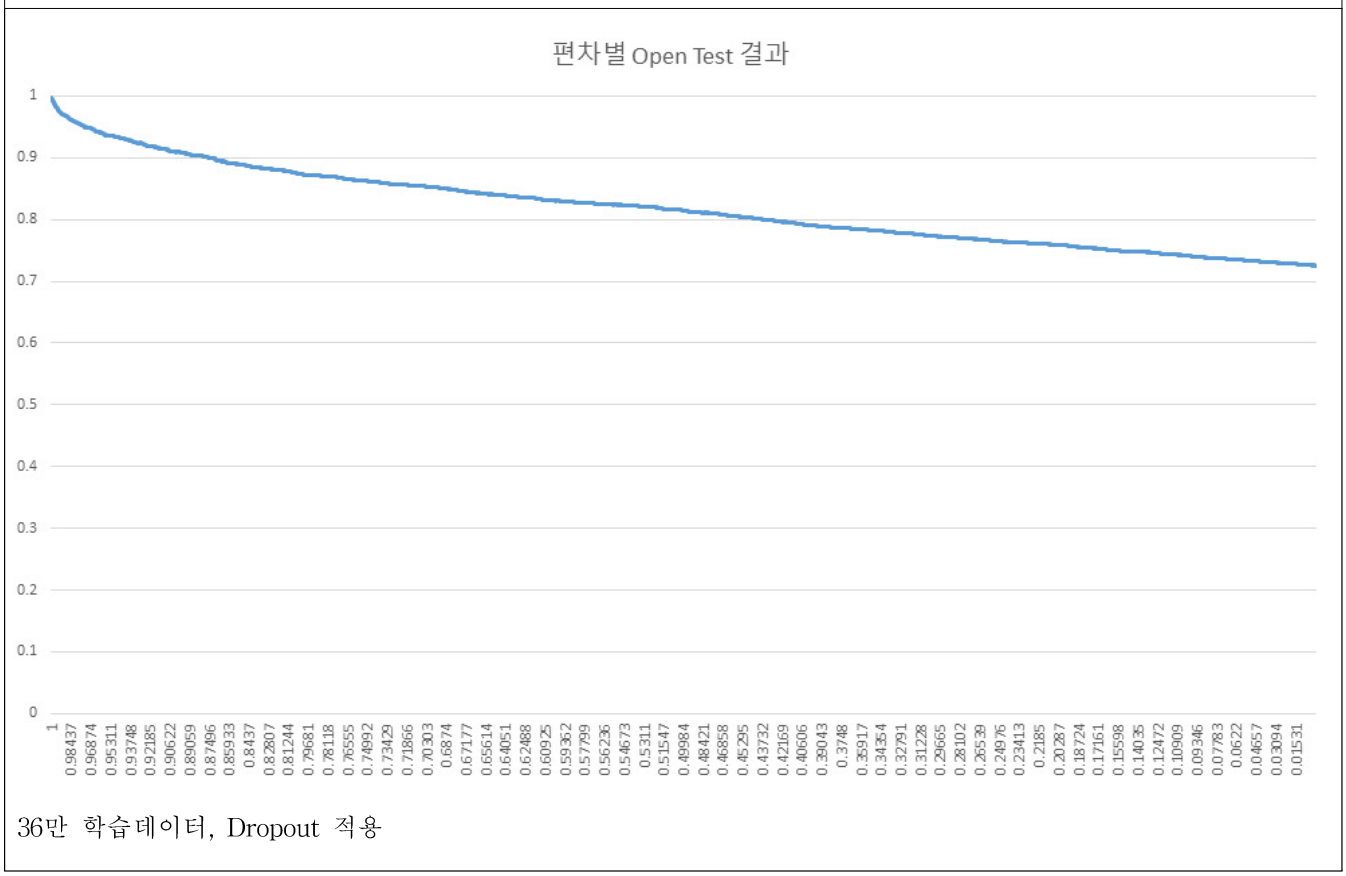


그림5 과적합문제 개선 후 평가 정확도 변화 그래프



8만 학습데이터, Dropout 미적용



36만 학습데이터, Dropout 적용

그림6 분류 편차별 분류 정확도 변화 그래프