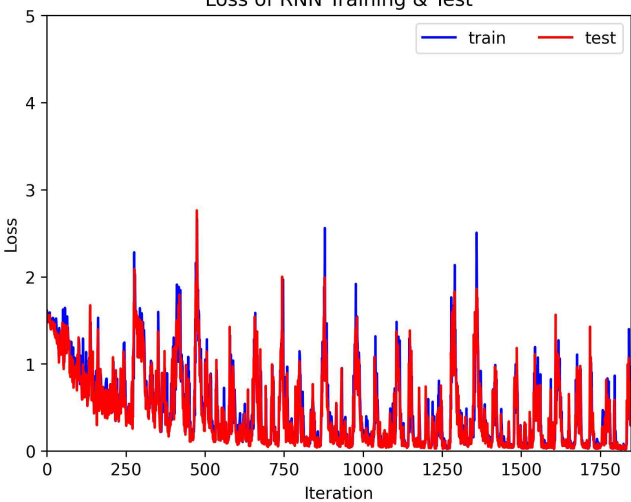
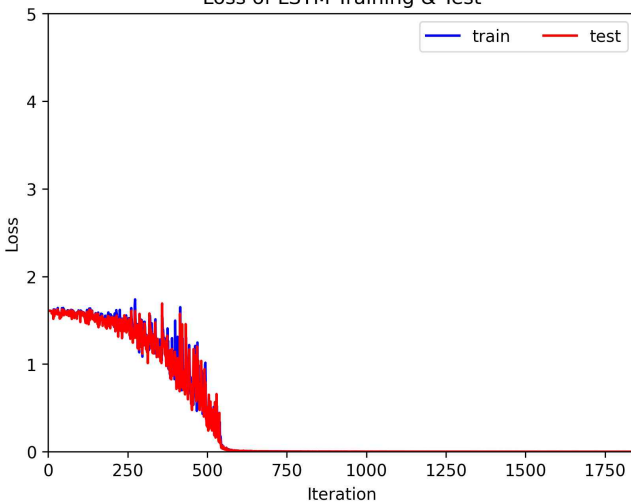
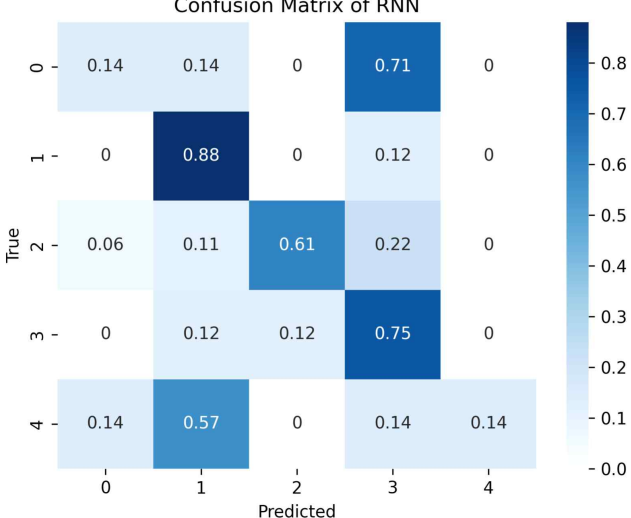
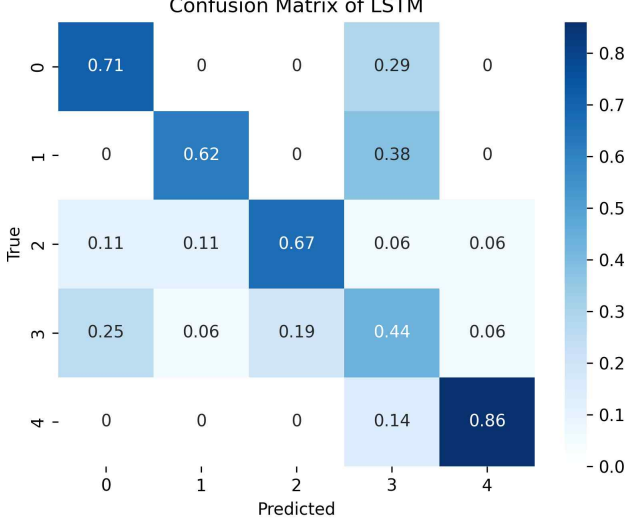




















































































































PA2 Report























































201811147 이지환

1. Result Comparison in terms of Structures

	RNN Structures	LSTM Structures
Parameter Setting	Model: RNN Optimizer: SGD Learning Rate: 1e-4 Vector Embedding: 50d Batch size: 50 Max Epoch: 700 Dropout Rate: 0	Model: LSTM Optimizer: SGD Learning Rate: 1e-2 Vector Embedding: 50d Batch size: 50 Max Epoch: 700 Dropout Rate: 0
Traning Loss Graph	 <p>Loss of RNN Training & Test</p>	 <p>Loss of LSTM Training & Test</p>
Complex Matrix	 <p>Confusion Matrix of RNN</p>	 <p>Confusion Matrix of LSTM</p>

All Emojis for Test Set			
Test String	True Value	RNN Answer	LSTM Answer
I want to eat			
he did not answer			
he got a very nice raise			
she got me a nice present			

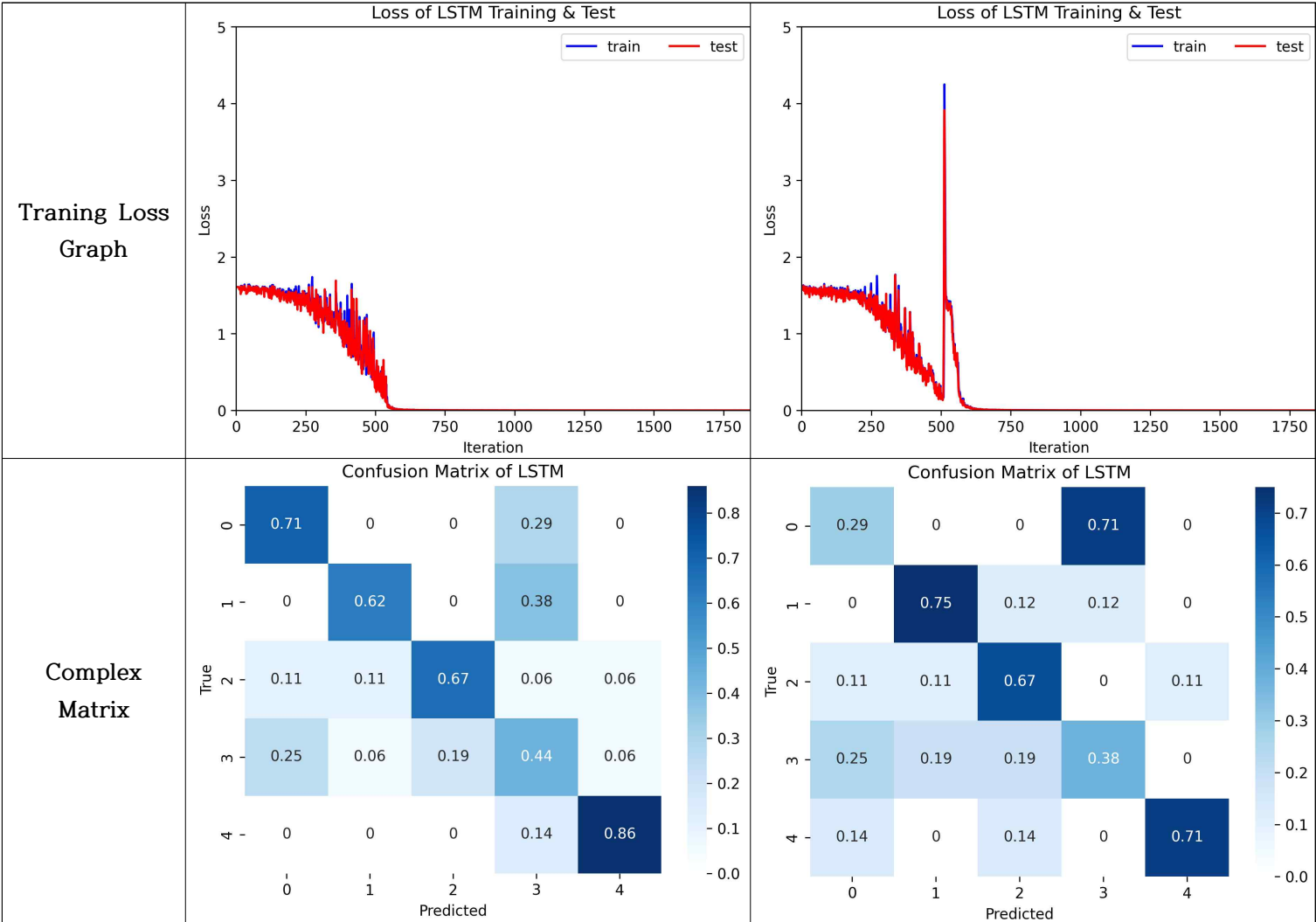
ha ha ha it was so funny			
he is a good friend			
I am upset			
We had such a lovely dinner tonight			
where is the food			
Stop making this joke ha ha ha			
where is the ball			
work is hard			
This girl is messing with me			
are you serious			
Let us go play baseball			
This stupid grader is not working			
work is horrible			
Congratulation for having a baby			
stop pissing me off			
any suggestions for dinner			
I love taking breaks			
you brighten my day			
I boiled rice			
she is a bully			
Why are you feeling bad			
I am upset			
give me the ball			
My grandmother is the love of my life			
enjoy your game			
valentine day is near			
I miss you so much			
throw the ball			
My life is so boring			
she said yes			
will you be my valentine			
he can pitch really well			
dance with me			
I am hungry			



See you at the restaurant			
I like to laugh			
I will run			
I like your jacket			
i miss her			
what is your favorite baseball game			
Good job			
I love you to the stars and back			
What you did was awesome			
ha ha ha lol			
I do not want to joke			
go away			
yesterday we lost again			
family is all I have			
you are failing this exercise			
Good joke			
You deserve this nice prize			
I did not have breakfast			












이 결과의 Training Loss Graph를 보면, RNN은 Iteration이 진행됨에 따라 점차 Converge하는 경향을 가지지만 지속적으로 큰 Noise가 발생하는 것을 확인할 수 있다. 반면에 LSTM은 Iteration이 550 정도일 때부터 큰 Noise 없이 안정적으로 Converge하는 경향을 확인할 수 있다. 이러한 Converge의 경향성은 Test Data Set에 대한 Prediction 결과의 정확도에도 반영되어 LSTM이 평균 정확도 66%으로 평균 정확도 50.4%인 RNN보다 약 16% 정도 더 높은 정확도를 보이는 것을 확인할 수 있다. 이는 RNN Model이 Model 특성 상 시계열 데이터에서 앞에 위치한 정보가 충분히 뒤로 전달되지 못하는 단점을 가지고 있어서 가장 중요한 정보가 앞쪽의 시점에 위치할 경우 앞쪽의 중요한 정보가 뒤로 오면서 정보량의 손실이 일어나기 때문에 일어나는 문제로 분석할 수 있다. 반면 LSTM은 Model에서 시계열 데이터에서 앞에 위치한 정보의 중요도를 판단해서 충분히 뒤로 전달할 수 있도록 Model이 Design되어 있기 때문에 더 높은 정확도를 보이고 더 안정적으로 Converge하는 것으로 해석할 수 있다.

2. Result Comparison in terms of Length of Glove Vectors

	LSTM with 50d Vector Embedding	LSTM with 100d Vector Embedding
Parameter Setting	Model: LSTM Optimizer: SGD Learning Rate: 1e-2 Vector Embedding: 50d Batch size: 50 Max Epoch: 700 Dropout Rate: 0	Model: LSTM Optimizer: SGD Learning Rate: 1e-2 Vector Embedding: 100d Batch size: 50 Max Epoch: 700 Dropout Rate: 0



All Emojis for Test Set			
Test String	True Value	50d LSTM Answer	100d LSTM Answer
I want to eat			
he did not answer			
he got a very nice raise			
she got me a nice present			
ha ha ha it was so funny			
he is a good friend			
I am upset			
We had such a lovely dinner tonight			
where is the food			
Stop making this joke ha ha ha			
where is the ball			
work is hard			
This girl is messing with me			
are you serious			

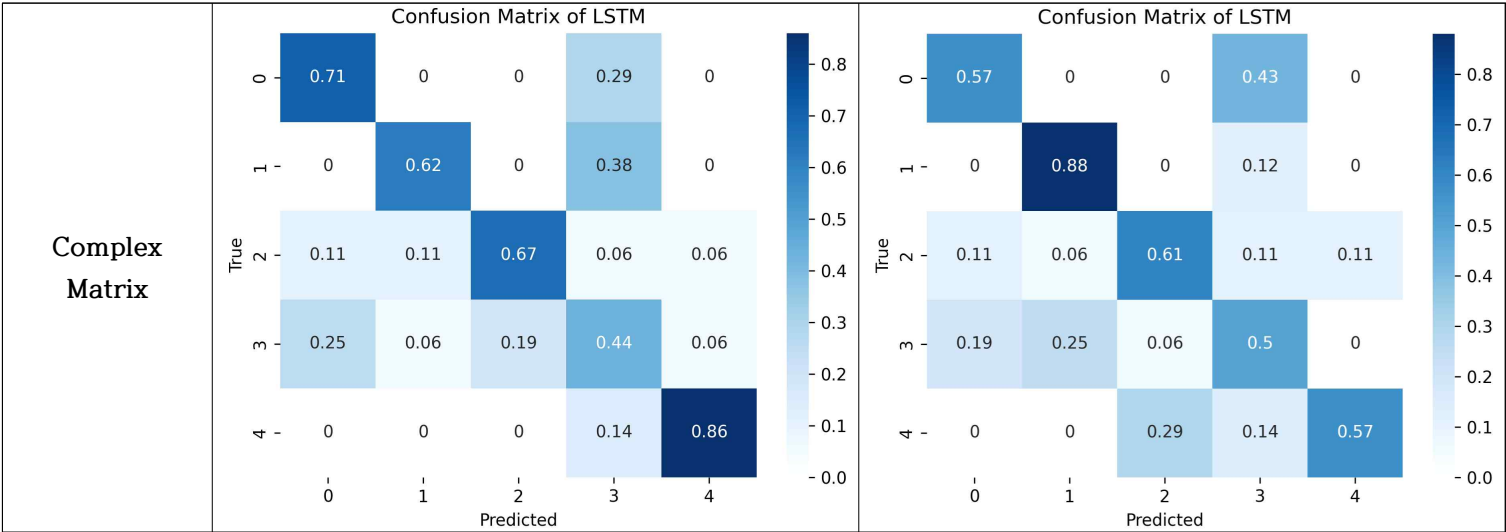
Let us go play baseball			
This stupid grader is not working			
work is horrible			
Congratulation for having a baby			
stop pissing me off			
any suggestions for dinner			
I love taking breaks			
you brighten my day			
I boiled rice			
she is a bully			
Why are you feeling bad			
I am upset			
give me the ball			
My grandmother is the love of my life			
enjoy your game			
valentine day is near			
I miss you so much			
throw the ball			
My life is so boring			
she said yes			
will you be my valentine			
he can pitch really well			
dance with me			
I am hungry			
See you at the restaurant			
I like to laugh			
I will run			
I like your jacket			
i miss her			
what is your favorite baseball game			
Good job			
I love you to the stars and back			
What you did was awesome			
ha ha ha lol			

I do not want to joke	😞	❤️	😄
go away	😞	😄	😄
yesterday we lost again	😞	🏈	🏈
family is all I have	❤️	😞	😞
you are failing this exercise	😞	😞	😞
Good joke	😄	😄	😄
You deserve this nice prize	😄	🏈	🏈
I did not have breakfast	🍴🍴	😞	❤️








이 결과의 Traning Loss Graph를 보면 Word를 50d Vector로 Embedding한 경우와 Word를 100d Vector로 Embedding한 경우가 전부 안정적으로 Converge하고 있는 것을 확인할 수 있다. 다만, Word를 50d Vector로 Embedding한 경우에는 비교적 크게 진동하면서 Converge하고 있고 Word를 100d Vector로 Embedding한 경우에는 비교적 적게 진동하면서 Converge하고 있는 것을 확인할 수 있다. 이는 Word를 100d Vector로 Embedding한 경우가 50d Vector로 Embedding한 경우보다 더 Word 자체가 가진 많은 정보량을 표현할 수 있기 때문일 것이라고 해석할 수 있다. 다만, Word Embedding을 50d Vector에서 100d Vector로 키우면 그만큼 Model에서 Learning해야 하는 Parameter의 개수가 증가하기 때문에 이를 위해 더 많은 양의 Traning Data가 필요하고 Converge하는데에도 더 많은 Iteration 횟수가 된다. 하지만, 위에서는 Word Embedding을 50d Vector로 한 경우와 100d Vector로 한 경우 모두 동일한 132개의 Data로 Training을 했기 때문에 Word Embedding을 100d Vector로 한 경우가 Learning을 하기에 Traing Data의 개수가 충분하지 않아 Word Embedding을 50d Vector로 한 경우보다 약 10%정도 더 낮은 정확도를 보이는 것을 확인할 수 있다.

3. Result Comparison in terms of Dropout

	LSTM without Dropout	LSTM with Dropout
Parameter Setting	Model: LSTM Optimizer: SGD Learning Rate: 1e-2 Vector Embedding: 50d Batch size: 50 Max Epoch: 700 Dropout Rate: 0	Model: LSTM Optimizer: SGD Learning Rate: 1e-2 Vector Embedding: 50d Batch size: 50 Max Epoch: 1000 Dropout Rate: 0.3
Traning Loss Graph	<div>Loss of LSTM Training & Test</div>	<div>Loss of LSTM Training & Test</div>



All Emojis for Test Set			
Test String	True Value	LSTM without dropout Answer	LSTM with dropout Answer
I want to eat			
he did not answer			
he got a very nice raise			
she got me a nice present			
ha ha ha it was so funny			
he is a good friend			
I am upset			
We had such a lovely dinner tonight			
where is the food			
Stop making this joke ha ha ha			
where is the ball			
work is hard			
This girl is messing with me			
are you serious			
Let us go play baseball			
This stupid grader is not working			
work is horrible			
Congratulation for having a baby			
stop pissing me off			
any suggestions for dinner			
I love taking breaks			

you brighten my day			
I boiled rice			
she is a bully			
Why are you feeling bad			
I am upset			
give me the ball			
My grandmother is the love of my life			
enjoy your game			
valentine day is near			
I miss you so much			
throw the ball			
My life is so boring			
she said yes			
will you be my valentine			
he can pitch really well			
dance with me			
I am hungry			
See you at the restaurant			
I like to laugh			
I will run			
I like your jacket			
i miss her			
what is your favorite baseball game			
Good job			
I love you to the stars and back			
What you did was awesome			
ha ha ha lol			
I do not want to joke			
go away			
yesterday we lost again			
family is all I have			
you are failing this exercise			
Good joke			
You deserve this nice prize			

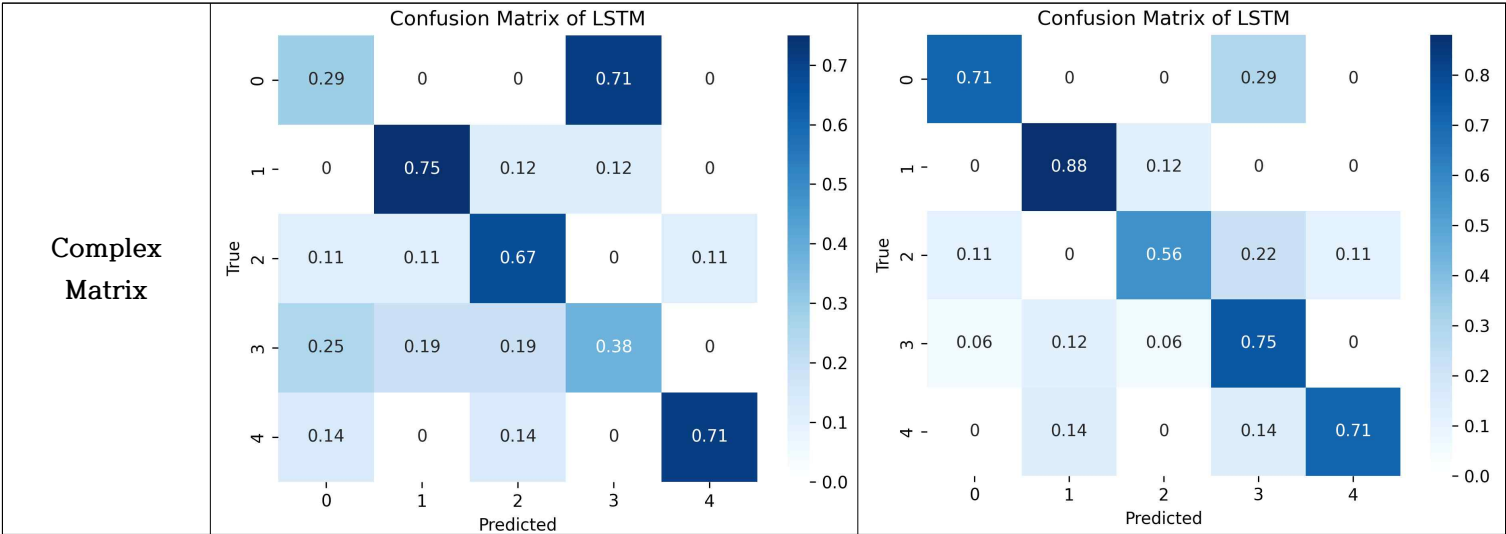
I did not have breakfast











이 결과의 Training Loss Graph를 보면 Dropout을 한 경우가 Dropout을 하지 않은 경우보다 약 10 Iteration 정도 더 늦게 Converge하고 있는 것을 확인할 수 있다. 이는 Dropout을 하게 되면 그만큼 한 Iteration에서 사용되는 Node의 개수가 적기 때문에 모든 Node들을 충분히 Learning시키기 위해서는 더 많은 Iteration을 돌아야 하기 때문으로 해석할 수 있다. 이러한 Dropout을 Learning을 할 때 적용하는 이유는 Overfitting을 막기 위해서인데, 위에서 진행한 Training은 Training Data의 개수가 132개로 너무 적기 때문에 Overfitting의 위험성이 크지만 Dropout을 하지 않아도 크게 Overfitting 되지 않은 것을 확인할 수 있다. 그래서 Dropout을 한 결과와 크게 차이가 있는 결과는 볼 수 없었다. Dropout을 한 경우가 평균 정확도 62.6%으로, Dropout을 하지 않은 경우의 평균 정확도인 66%보다 평균 정확도가 약 4% 정도 더 낮아지기는 했지만 Dropout을 한 경우가 Dropout을 하지 않은 경우보다 더 가중치가 큰 Feature가 생길 확률이 적고 이로 인해 모든 Node들이 골고루 최적화되어 가중치가 큰 Feature에 의해 Overfitting이 되지 않게 Training 되었을 것이라 유추해볼 수 있다.

4. Result Comparison in terms of Optimizers

	LSTM with SGD Optimizer	LSTM with ADAM Optimizer
Parameter Setting	Model: LSTM Optimizer: SGD Learning Rate: 1e-2 Vector Embedding: 100d Batch size: 50 Max Epoch: 700 Dropout Rate: 0	Model: LSTM Optimizer: ADAM beta1 = 0.9 beta2 = 0.99 epsilon = 1e-8 Learning Rate: 1e-3 Vector Embedding: 100d Batch size: 50 Max Epoch: 700 Dropout Rate: 0
Traning Loss Graph		



All Emojis for Test Set			
Test String	True Value	LSTM with SGD Answer	LSTM with ADAM Answer
I want to eat			
he did not answer			
he got a very nice raise			
she got me a nice present			
ha ha ha it was so funny			
he is a good friend			
I am upset			
We had such a lovely dinner tonight			
where is the food			
Stop making this joke ha ha ha			
where is the ball			
work is hard			
This girl is messing with me			
are you serious			
Let us go play baseball			
This stupid grader is not working			
work is horrible			
Congratulation for having a baby			
stop pissing me off			
any suggestions for dinner			
I love taking breaks			

you brighten my day			
I boiled rice			
she is a bully			
Why are you feeling bad			
I am upset			
give me the ball			
My grandmother is the love of my life			
enjoy your game			
valentine day is near			
I miss you so much			
throw the ball			
My life is so boring			
she said yes			
will you be my valentine			
he can pitch really well			
dance with me			
I am hungry			
See you at the restaurant			
I like to laugh			
I will run			
I like your jacket			
i miss her			
what is your favorite baseball game			
Good job			
I love you to the stars and back			
What you did was awesome			
ha ha ha lol			
I do not want to joke			
go away			
yesterday we lost again			
family is all I have			
you are failing this exercise			
Good joke			
You deserve this nice prize			

I did not have breakfast



이 결과의 Training Loss Graph를 보면 SGD Optimizer를 활용한 경우보다 ADAM Optimizer를 활용한 경우가 더 빠르게 Converge하고 있는 것을 확인할 수 있다. 이는 SGD는 모든 Step에 대해 동일한 가중치를 주기 때문에 Converge에 가까워져도 안정적으로 Converge하기보다는 일정한 보폭으로 Converge에 다가가는 반면 ADAM은 진행하던 속도에 관성을 주고 최근 경로의 gradient에 따른 적절한 learning rate를 계산하여 반영하기 때문에 보다 빠르게 Converge할 수 있는 지점을 찾기 때문이라고 분석할 수 있다. 또한 SGD는 Converge해도 Converge한 Loss값이 0.09 정도로 $1e-5$ Loss값에 Converge하는 ADAM보다 비교적 더 큰데 이는 앞서서도 언급했듯 SGD는 모든 Step에 대해 동일한 가중치를 주기 때문에 보다 낮은 Loss 지점을 찾아내기 힘든 Optimizer인 반면에 ADAM은 최근 경로의 gradient에 따른 적절한 learning rate를 계산하여 반영하기 때문에 보다 낮은 Loss 지점을 계속해서 찾기 때문에 더 낮은 Loss에서 Converge한다는 것으로 해석할 수 있다. 그렇기 때문에 ADAM Optimizer를 사용한 경우가 평균 정확도 72.2%로, 평균 정확도가 56%인 SGD Optimizer를 사용한 경우와 비교할 때 적은 Training Data 개수를 사용함에도 약 16%나 더 높은 정확도로 Prediction을 할 수 있게 되었다고 분석할 수 있다. 다만, 위의 Training은 SGD와 ADAM을 서로 비교하기 위해서 Batch Size를 50으로 설정하여 진행하였기 때문에 낮게 최적화한 Loss를 더 낮게 최적화하게 동작한 결과 어떤 Test Data에 대해서는 거의 100%의 Prediction으로 Class 판단을 하는 반면 어떤 Test Data에 대해서는 40%의 Prediction으로 Class 판단을 하는 등 Model이 Class Prediction에 대해 확신하는 Percentage가 차이가 나게 되었다. 이러한 문제 때문에 Batch Size를 보다 작게 설정하여 ADAM Optimizer를 돌리면 보다 높은 정확도로 Class Prediction을 할 수 있게 Training할 수 있을 것이라 생각한다. 반면 SGD는 Batch Size를 줄이게 되면 보다 Noise가 많이 발생하여 Converge하기 더 어려워질 수 있기 때문에 learning rate도 함께 고려하면서 Batch Size를 줄여야 할 것이라고 생각한다.

5. Difference between Word2Vec and Glove in terms of Vector Generation

Word2Vec과 Glove 모두 Word를 Vector로 바꾸는 Embedding 방법들이다. 우선 Word2Vec은 같은 문장에서 나타난 인접한 단어들 사이에는 의미가 비슷할 것이라는 Distributional Hypothesis에 근거해서 두 가지 방법으로 나눠서 Word Embedding을 학습시킨다. 이 두 가지 방법은 CBOW와 Skip-Gram인데, CBOW는 target이 되는 가운데에 있는 단어를 기준으로 좌우에 있는 단어들을 몇 개까지 볼 건지 미리 정한 window를 입력으로 가운데에 있는 단어를 예측하면서 학습을 진행하고, Skip-Gram은 target이 되는 가운데에 있는 단어를 기준으로 window 범위에 있는 단어들을 예측하면서 학습을 진행한다. 그리고 이렇게 학습시킨 Embedding을 Word를 Vector로 바꿔서 Learning을 진행해야 하는 Deep Learning에 활용한다. 반면에 Glove는 학습 데이터에서 어떤 단어쌍이 동시에 등장한 횟수를 미리 계산하고 이에 대한 로그 값을 두 단어간의 내적값과 큰 차이가 나지 않도록 Word Embedding을 학습시키고 이렇게 학습시킨 Embedding을 Word를 Vector로 바꿔서 Learning을 진행해야 하는 Deep Learning에 활용한다. 요약하면, Word2Vec은 사용자가 지정한 주변 단어의 개수에 기반해서 Word Embedding을 학습하는 반면에, Glove는 전체 학습 데이터에서 단어쌍의 동시 등장 횟수에 기반해서 Word Embedding을 학습한다고 요약할 수 있다. 이렇게 Word Embedding을 학습시키기 때문에 Word2Vec 방식은 학습 데이터 전체에 대한 정보를 담기 어려운 반면, Glove는 단어 간의 문법적인 의미와 관계들까지 효과적으로 학습시킬 수 있다는 차이가 있다.