

**Nama** : Jiddy Abdillah  
**NIM** : 1301162765  
**Kelas** : SIDE-40-GAB Penambangan Teks

## Analisis Performa BiLSTM dan CNN menggunakan IMDB Review Dataset

### A. Strategi Pendekatan Masalah

Pada penyelesaian masalah klasifikasi teks, ada beberapa komponen penting, yaitu:

#### 1. *Training Text and Labels*

Pada masalah ini saya menggunakan dataset IMDB berisi review sebuah film dan label sentiment dari review tersebut. Jumlah data pada dataset ini yaitu sebanyak 25000 data.

	id	sentiment	review
0	5814_8	1	With all this stuff going down at the moment w...
1	2381_9	1	\The Classic War of the Worlds\" by Timothy Hi...
2	7759_3	0	The film starts with a manager (Nicholas Bell)...

#### 2. *Feature Vector*

*Feature Vector* yang digunakan pada masalah ini adalah GloVe 6B 100d, berisi 6 miliar kata dan memiliki 100 dimensi yang nantinya digunakan pada layer embedding dari model prediktif yang dibuat. Fungsi *feature vector* yaitu mengambil vektor representasi dari sebuah kata.

#### 3. *Learning Algorithm*

*Learning algorithm* yang digunakan pada masalah ini adalah simplified *Convolutional Neural Network* (CNN) dan *Bidirectional Long Short Term Memory* (BiLSTM).

CNN adalah kelas jaringan saraf tiruan dimana koneksi antara *nodes* tidak membentuk suatu siklus, sedangkan BiLSTM membentuk koneksi graf arah bersama sekuens antara *nodes* yang ada. BiLTSM dilatih untuk mengenal pattern berdasarkan waktu sedang CNN dilatih untuk mengenal pattern berdasarkan ruang.

#### 4. *Predictive Model*

Model yang dibuat terdiri dari empat layer utama, yaitu *input layer*, *embedding layer*, *dropout layer*, dan *dense layer*. Embedding layer menggunakan word2vec yang sudah ada yaitu GloVe. Nilai *dropout* yang digunakan pada model yaitu 0,5.

Model CNN yang digunakan yaitu model satu dimensi. Tambahan layer pada model CNN yaitu menggunakan layer *flatten* dan tiga layer *max pooling*.

## B. Output Program

```
Epoch 1/10  
20000/20000 [=====] - 943s 47ms/step - loss: 0.5066 - acc: 0.7558 - val_loss: 0.3919 - val_acc: 0.8270  
Epoch 00001: val_acc improved from -inf to 0.82700, saving model to model_rnn.hdf5  
Epoch 2/10  
20000/20000 [=====] - 947s 47ms/step - loss: 0.3433 - acc: 0.8572 - val_loss: 0.2982 - val_acc: 0.8838  
Epoch 00002: val_acc improved from 0.82700 to 0.88380, saving model to model_rnn.hdf5  
Epoch 3/10  
20000/20000 [=====] - 948s 47ms/step - loss: 0.2611 - acc: 0.8988 - val_loss: 0.3933 - val_acc: 0.8804  
Epoch 00003: val_acc did not improve from 0.88380  
Epoch 4/10  
20000/20000 [=====] - 946s 47ms/step - loss: 0.2895 - acc: 0.9195 - val_loss: 0.2921 - val_acc: 0.8936  
Epoch 00004: val_acc improved from 0.88380 to 0.89360, saving model to model_rnn.hdf5  
Epoch 5/10  
20000/20000 [=====] - 950s 47ms/step - loss: 0.1718 - acc: 0.9366 - val_loss: 0.2782 - val_acc: 0.8954  
Epoch 00005: val_acc improved from 0.89360 to 0.89540, saving model to model_rnn.hdf5  
Epoch 6/10  
20000/20000 [=====] - 951s 48ms/step - loss: 0.1428 - acc: 0.9499 - val_loss: 0.3044 - val_acc: 0.8964  
Epoch 00006: val_acc improved from 0.89540 to 0.89640, saving model to model_rnn.hdf5  
Epoch 7/10  
20000/20000 [=====] - 949s 47ms/step - loss: 0.1161 - acc: 0.9682 - val_loss: 0.3065 - val_acc: 0.8966  
Epoch 00007: val_acc improved from 0.89640 to 0.89660, saving model to model_rnn.hdf5  
Epoch 8/10  
20000/20000 [=====] - 953s 48ms/step - loss: 0.0965 - acc: 0.9677 - val_loss: 0.3122 - val_acc: 0.8932  
Epoch 00008: val_acc did not improve from 0.89660  
Epoch 9/10  
20000/20000 [=====] - 954s 48ms/step - loss: 0.0739 - acc: 0.9751 - val_loss: 0.3464 - val_acc: 0.8860  
Epoch 00009: val_acc did not improve from 0.89660  
Epoch 10/10  
20000/20000 [=====] - 952s 48ms/step - loss: 0.0575 - acc: 0.9815 - val_loss: 0.3526 - val_acc: 0.8906  
Epoch 00010: val_acc did not improve from 0.89660
```

Gambar 1. BiLSTM

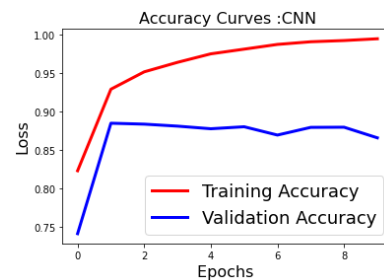
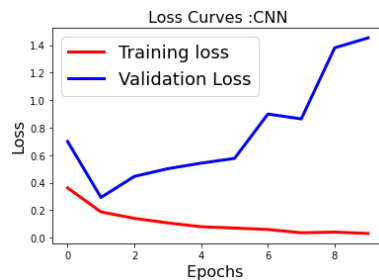
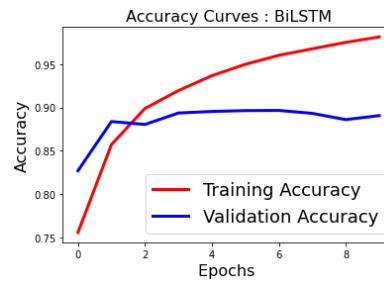
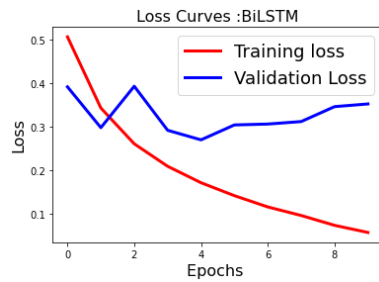
```
Epoch 1/10  
20000/20000 [=====] - 267s 13ms/step - loss: 0.3618 - acc: 0.8232 - val_loss: 0.6993 - val_acc: 0.7414  
Epoch 00001: val_acc improved from -inf to 0.74140, saving model to model_rnn.hdf5  
Epoch 2/10  
20000/20000 [=====] - 271s 14ms/step - loss: 0.1874 - acc: 0.9292 - val_loss: 0.2924 - val_acc: 0.8850  
Epoch 00002: val_acc improved from 0.74140 to 0.88500, saving model to model_rnn.hdf5  
Epoch 3/10  
20000/20000 [=====] - 264s 13ms/step - loss: 0.1405 - acc: 0.9518 - val_loss: 0.4452 - val_acc: 0.8838  
Epoch 00003: val_acc did not improve from 0.88500  
Epoch 4/10  
20000/20000 [=====] - 267s 13ms/step - loss: 0.1075 - acc: 0.9642 - val_loss: 0.5814 - val_acc: 0.8812  
Epoch 00004: val_acc did not improve from 0.88500  
Epoch 5/10  
20000/20000 [=====] - 265s 13ms/step - loss: 0.0794 - acc: 0.9752 - val_loss: 0.5418 - val_acc: 0.8778  
Epoch 00005: val_acc did not improve from 0.88500  
Epoch 6/10  
20000/20000 [=====] - 267s 13ms/step - loss: 0.0700 - acc: 0.9813 - val_loss: 0.5761 - val_acc: 0.8804  
Epoch 00006: val_acc did not improve from 0.88500  
Epoch 7/10  
20000/20000 [=====] - 265s 13ms/step - loss: 0.0595 - acc: 0.9875 - val_loss: 0.8987 - val_acc: 0.8696  
Epoch 00007: val_acc did not improve from 0.88500  
Epoch 8/10  
20000/20000 [=====] - 265s 13ms/step - loss: 0.0355 - acc: 0.9909 - val_loss: 0.8635 - val_acc: 0.8796  
Epoch 00008: val_acc did not improve from 0.88500  
Epoch 9/10  
20000/20000 [=====] - 269s 13ms/step - loss: 0.0407 - acc: 0.9925 - val_loss: 1.3801 - val_acc: 0.8798  
Epoch 00009: val_acc did not improve from 0.88500  
Epoch 10/10  
20000/20000 [=====] - 263s 13ms/step - loss: 0.0304 - acc: 0.9948 - val_loss: 1.4524 - val_acc: 0.8660  
Epoch 00010: val_acc did not improve from 0.88500
```

Gambar 2. CNN

Pada BiLSTM, hasil setelah 10 iterasi yaitu memiliki nilai *training accuracy* terbesar dan *loss* terkecil yaitu 98% dan 0,0575. *Validation accuracy* terbesar dan *loss* terkecil yaitu 89% dan 0,2702.

Pada CNN, hasil setelah 10 iterasi yaitu memiliki nilai *training accuracy* terbesar dan *loss* terkecil yaitu 99% dan 0,0304. *Validation accuracy* terbesar dan *loss* terkecil yaitu 88% dan 1,4524.

## C. Analisis



Dari hasil grafik diatas, perbedaan terbesar terjadi kepada *validation loss* model CNN. Semakin banyak iterasi, nilai *validation loss* semakin besar. Nilai *validation loss* lebih besar dari *accuracy loss* terjadi karena jumlah data yang lebih sedikit. Untuk mengatasi nilai *loss* yang terlalu besar, dapat dilakukan pengaturan *dropout rate*, mengurangi kompleksitas dari model dan menentukan *learning rate* dan *decay rate*.