KLASIFIKASI ULASAN FILM DI IMDB MENGGUNAKAN METODE CNN

Disusun sebagai syarat UAS
mata kuliah *Machine Learning*

Kevin Andrew A. J. W. – 140810160012
Adryan Luthfi Faiz – 140810160049
Patricia Joanne – 140810160065

Dikumpulkan tanggal
16 Desember 2019

PROGRAM STUDI S-1 TEKNIK INFORMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS PADJADJARAN
2019

BAB I
PENDAHULUAN

1.1 Latar Belakang

Informasi merupakan suatu kebutuhan dan sampai saat ini bermunculan sangat tidak terbatas jumlahnya sehingga dibutuhkan suatu penyajian dari informasi tanpa mengurangi nilai yang terkandung di dalamnya. Dengan berkembangnya dunia maya di internet, semakin banyak orang yang menuliskan opini mereka tentang sebuah produk atau jasa. Dalam beberapa tahun terakhir, sejumlah besar situs telah memungkinkan penggunanya untuk berkontribusi, memodifikasi, dan meningkatkan konten. Setiap pengguna internet aktif dengan bebas menyampaikan ekspresi mereka atau pendapat pribadi tentang suatu topik tertentu, salah satu contohnya yaitu film.

Dengan kemajuan teknologi yang sangat pesat sekarang ini, seluruh informasi tentang film sudah tersedia di internet. Semakin banyak informasi yang disediakan di internet, maka akan semakin sulit juga untuk menemukan informasi yang sesuai dengan kebutuhan konsumen. Salah satu hal yang penting untuk menilai sebuah film adalah ulasan yang merupakan penilaian seseorang berupa opini. Opini yang terkandung dalam ulasan bisa positif atau negatif, maka dari itu dibutuhkan sebuah pengklasifikasian ulasan yang dapat membedakan opini positif dan opini negatif.

IMDb yang merupakan singkatan dari *Internet Movie Database* dan alamat situs www.imdb.com adalah situs yang dibuat untuk memuat berbagai informasi tentang film, program televisi dan *video game*, termasuk aktor, *crew*, dan karakter fiksi yang terdapat dalam tiga media hiburan visual tersebut. IMDb diluncurkan pada tanggal 17 Oktober 1990 dan pada tahun 1998 diakuisisi oleh Amazon.com. Terdapat pula komunitas di situs IMDb yang berkontribusi langsung untuk menuangkan ulasan tentang film dan memberikan rating pada ketiga media hiburan visual yang telah disebutkan di atas tersebut. Tidak hanya kaum awam, para pakarpun juga mempunyai wadah sendiri untuk memberi rating dan menuangkan ulasan secara profesional.

Jika kita mengunjungi salah satu laman film di IMDb, kita akan menemukan sesuatu yang disebut *IMDb user rating*, yakni semacam nilai seberapa bagus film tersebut di mata para pengguna IMDB yang lain Pengguna yang telah terdaftar dapat memberikan penilaiannya atas suatu film.

Dengan menjadikan IMDb sebagai sumber data dari ulasan film yang akan kami gunakan, kami akan membuat sebuah program untuk mengklasifikasikan ulasan film apakah film tersebut dinilai sebagai film yang bagus atau buruk menggunakan salah satu metode *deep learning* yaitu CNN. CNN sesuai dengan namanya, memanfaatkan proses konvolusi dan proses ini cukup sesuai untuk klasifikasi teks seperti ulasan film.

1.2 Rumusan Masalah

Berdasarkan latar belakang tersebut, dapat dirumuskan masalah sebagai berikut:

1. Bagaimana cara mengelompokkan ulasan yang positif dan negatif dari sebuah film?
2. Bagaimana cara mengetahui bagaimana penilaian secara umum oleh penonton dari sebuah film?

1.3 Manfaat dan Tujuan

Program yang dibuat memiliki manfaat dan tujuan sebagai berikut:

1. Menerapkan metode CNN pada klasifikasi ulasan film.
2. Menganalisis performa dari hasil klasifikasi dokumen.

BAB II
PEMBAHASAN

2.1 Metode yang Digunakan

Metode yang digunakan dalam program ini adalah salah satu metode dari *deep learning* yaitu CNN (*Convolutional Neural Network*). *Convolutional neural network* (CNN) dapat diterapkan untuk melakukan klasifikasi dokumen teks. Seiring dengan berkembangnya komputasi menggunakan *Graphical Processing Unit* (GPU) membuat proses pelatihan model pada algoritma CNN juga menjadi lebih cepat.

Cara kerja CNN memiliki kesamaan pada MLP (*Multilayer Perceptron*), namun dalam CNN setiap neuron atau input dari *layer* sebelumnya dipresentasikan dalam bentuk dua dimensi, tidak seperti MLP yang setiap neuron hanya berukuran satu dimensi. CNN merupakan pengembangan lebih lanjut dari MLP karena menggunakan metode yang mirip dengan dimensi yang lebih banyak. Di algoritma CNN ini, input dari layer sebelumnya bukan array 1 dimensi melainkan array 2 dimensi.

2.2 Dataset yang Digunakan

Dataset yang digunakan dalam program ini adalah kumpulan *review* film di situs IMDb. Dataset disadur dari https://keras.io/datasets/ yang berisi 25.000 ulasan film dari IMDb dan telah dilabeli oleh sentimen positif dan negatif (dapat diunduh di http://ai.stanford.edu/%7Eamaas/data/sentiment/). Ulasan-ulasan ini telah melalui *pre-processing* dan setiap ulasan dikodekan sebagai urutan indeks kata (integer). Untuk memudahkan, kata-kata diindeks oleh frekuensi keseluruhan dalam dataset, misalnya integer "3" mengkodekan kata ke-3 paling sering dalam data. Hal ini memungkinkan operasi penyaringan cepat seperti "hanya mempertimbangkan 10.000 kata paling umum teratas dan menghilangkan 20 kata paling umum teratas". Sebagai konvensi, "0" tidak berarti kata tertentu, tetapi digunakan untuk menyandikan kata yang tidak dikenal.

2.3 Spesifikasi Sistem yang Digunakan

Berikut ini adalah spesifikasi sistem yang digunakan:

1. *Hardware*
    - Laptop

- OS: Windows 10 Education
- RAM: 8 GB
- Memory: 1 TB
- GPU: Nvidia Geforce GTX 1050

2. *Online Tools*
   - Google CoLab (https://colab.research.google.com/)

2.4 Langkah Pengaplikasian

Berikut langkah yang dilakukan selama perancangan program:

1. Pengumpulan data dan *pre-processing* data

   Hal pertama yang dilakukan adalah melakukan pengumpulan data, namun karena dataset yang kami gunakan adalah dataset yang telah disediakan oleh Keras.io, kami tidak perlu lagi melakukan pengumpulan data. Selain itu, data juga sudah melalui *pre-processing*.

   Cara pemanggilannya adalah sebagai berikut:

```
from keras.datasets import imdb

(x_train, y_train), (x_test, y_test) = imdb.load_data(path="imdb.npz",
                                                      num_words=None,
                                                      skip_top=0,
                                                      maxlen=None,
                                                      seed=113,
                                                      start_char=1,
                                                      oov_char=2,
                                                      index_from=3)
```

2. Pembuatan model CNN

   Dalam Keras terdapat dua cara dalam membangun model yaitu model *sequential* dan *function* API. *Function* API digunakan untuk menentukan model kompleks, sedangkan model *sequential* lebih seperti tumpukan linier dari *layer*. Model yang akan digunakan adalah model *sequential* karena model CNN mempunyai beberapa layer diantaranya *convolution*, *pooling* dan *dropout*.

- *Convolutional Layer* yang berguna untuk menghitung *output* dari neuron yang terhubung ke daerah lokal dalam input, masing-masing menghitung produk titik antara bobot mereka dan wilayah kecil yang terhubung ke dalam volume input.
- *Rectified Linear Unit* (ReLU) akan menghilangkan *vanishing gradient* dengan cara menerapkan fungsi aktivasi *element* sebagai f(x)=max(0,x) alias aktivasi elemen akan dilakukan saat berada di ambang batas 0.
- *Pooling Layer* yaitu lapisan yang mengurangi dimensi dari *feature map* sehingga mempercepat komputasi karena parameter yang harus diupdate semakin sedikit dan mengatasi *overfitting*. *Pooling* yang biasa digunakan adalah *Max Pooling* dan *Average Pooling*. *Max Pooling* untuk menentukan nilai maksimum tiap pergeseran filter, sementara *Average Pooling* akan menentukan nilai rata-ratanya.

3. Klasifikasi

   Tahap ini mengklasifikasikan tiap neuron yang telah diekstraksi fitur pada sebelumnya. Terdiri dari:

   - Flatten

   Membentuk ulang fitur (*reshape feature map*) menjadi sebuah *vector* agar bisa digunakan sebagai input dari *fully-connected layer*.

   - Fully-connected

   Lapisan FC akan menghitung skor kelas. Seperti jaringan saraf biasa dan sesuai dengan namanya, setiap neuron akan terhubung ke semua angka dalam volume.

   - Softmax

   Fungsi Softmax menghitung probabilitas dari setiap kelas target atas semua kelas target yang memungkinkan dan akan membantu untuk menentukan kelas target untuk input yang diberikan. Softmax menggunakan eksponensial (e-power) dari nilai input yang diberikan dan jumlah nilai eksponensial dari semua nilai dalam input.

4. Uji coba dan evaluasi

   Menggunakan Google CoLab, kami menjalankan program yang telah dibuat lalu melakukan beberapa iterasi untuk menguji model struktur mana yang paling tinggi akurasinya.

Berikut di bawah ini salah satu uji coba yang dilakukan pada program.

Aktivasi:

Conv.1 (Sigmoid)

Conv.2 (Sigmoid)

***Pooling*:**

Local (Average)

Global (Average)

```
Build model...
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:42

Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_2 (Embedding)      (None, 256, 50)           500000
_________________________________________________________________
dropout_3 (Dropout)          (None, 256, 50)           0
_________________________________________________________________
conv1d_2 (Conv1D)            (None, 252, 100)          25100
_________________________________________________________________
average_pooling1d_1 (Average (None, 50, 100)           0
_________________________________________________________________
conv1d_3 (Conv1D)            (None, 46, 100)           50100
_________________________________________________________________
global_average_pooling1d_2 ( (None, 100)               0
_________________________________________________________________
dense_3 (Dense)              (None, 200)               20200
_________________________________________________________________
dropout_4 (Dropout)          (None, 200)               0
_________________________________________________________________
dense_4 (Dense)              (None, 200)               40200
_________________________________________________________________
dropout_5 (Dropout)          (None, 200)               0
_________________________________________________________________
dense_5 (Dense)              (None, 1)                 201
=================================================================
Total params: 635,801
Trainable params: 635,801
Non-trainable params: 0
```

```
    results = model.fit(t_x, t_y, epochs=EPOCHS, batch_size=BATCH_SIZE, callbacks=[early_stopping, model_checkpoint],
              verbose=1, validation_split=0.1)
    print("Val Score: ", model.evaluate(val_x, val_y))
    return results
```

```
n_folds=10
epochs=5
batch_size=128

#simpan histori model dalam daftar setelah pemasangan sehingga kita dapat merencanakannya nanti
model_history = []

for i in range(n_folds):
    print("Training on Fold: ",i+1)
    # train = tokenizeX(reviews_train_clean)
    t_x, val_x, t_y, val_y = train_test_split(x_train, y_train, test_size=0.1,
                                              random_state = np.random.randint(1,1000, 1)[0])
    model_history.append(fit_and_evaluate(t_x, val_x, t_y, val_y, epochs, batch_size))
    print("=======*12, end="\n\n\n")
```

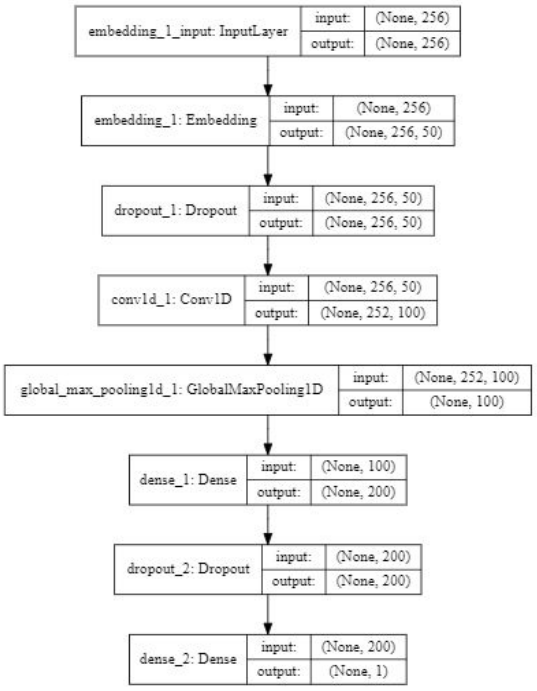
```
Training on Fold:  9
Train on 20250 samples, validate on 2250 samples
Epoch 1/5
20250/20250 [==============================] - 1s 55us/step - loss: 0.0983 - acc: 0.9698 - val_loss: 0.0914 - val_acc: 0.9720

Epoch 00001: val_loss improved from 0.10948 to 0.09136, saving model to fas_mnist_1.h5
Epoch 2/5
20250/20250 [==============================] - 1s 55us/step - loss: 0.0907 - acc: 0.9716 - val_loss: 0.1300 - val_acc: 0.9547

Epoch 00002: val_loss did not improve from 0.09136
Epoch 3/5

20250/20250 [==============================] - 1s 55us/step - loss: 0.0811 - acc: 0.9756 - val_loss: 0.1174 - val_acc: 0.9604

Epoch 00004: val_loss did not improve from 0.09136
Epoch 5/5
20250/20250 [==============================] - 1s 55us/step - loss: 0.0749 - acc: 0.9786 - val_loss: 0.1971 - val_acc: 0.9360

Epoch 00005: val_loss did not improve from 0.09136
2500/2500 [==============================] - 0s 44us/step
Val Score:  [0.21347394853830337, 0.928]
================================================================================


Training on Fold:  10
Train on 20250 samples, validate on 2250 samples
Epoch 1/5
20250/20250 [==============================] - 1s 56us/step - loss: 0.0843 - acc: 0.9739 - val_loss: 0.0797 - val_acc: 0.9791

Epoch 00001: val_loss improved from 0.09136 to 0.07971, saving model to fas_mnist_1.h5
Epoch 2/5
20250/20250 [==============================] - 1s 55us/step - loss: 0.0763 - acc: 0.9768 - val_loss: 0.1253 - val_acc: 0.9569

Epoch 00002: val_loss did not improve from 0.07971
Epoch 3/5
20250/20250 [==============================] - 1s 55us/step - loss: 0.0688 - acc: 0.9800 - val_loss: 0.1367 - val_acc: 0.9556

Epoch 00003: val_loss did not improve from 0.07971
Epoch 4/5
20250/20250 [==============================] - 1s 56us/step - loss: 0.0658 - acc: 0.9814 - val_loss: 0.3063 - val_acc: 0.9058

Epoch 00004: val_loss did not improve from 0.07971
Epoch 5/5
20250/20250 [==============================] - 1s 55us/step - loss: 0.0618 - acc: 0.9821 - val_loss: 0.1293 - val_acc: 0.9627

Epoch 00005: val_loss did not improve from 0.07971
2500/2500 [==============================] - 0s 48us/step
Val Score:  [0.12845237897187473, 0.9624]
================================================================================
```

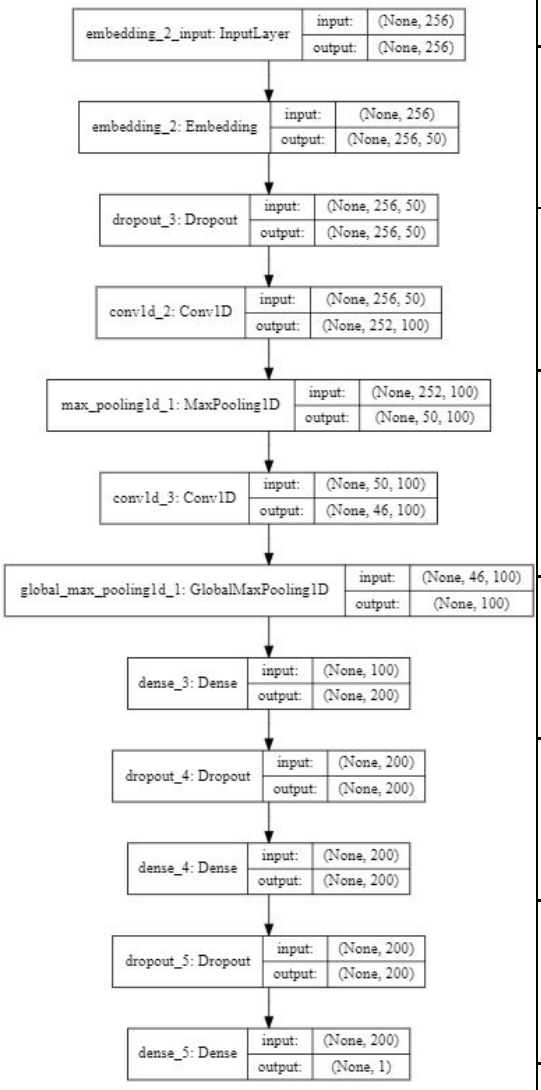
```
[18] scores = model.evaluate(x_test, y_test, verbose=2)
     print("%s: %.2f%%" % (model.metrics_names[1], scores[1]*100))
```

```
acc: 82.24%
```

BAB III
HASIL EKSPERIMEN

3.1 Tabel Hasil Eksperimen

No	Model Struktur	Aktivasi	Pooling	Akurasi %
1		Conv.1 (ReLu)	Global (Max)	87,04
		Conv.1 (Sigmoid)	Global (Max)	84,34
		Conv.1 (ReLu)	Global (Average)	82,98
		Conv.1 (Sigmoid)	Global (Average)	81,53

2		Conv.1 (ReLu) Conv.2 (ReLu)	Local (Max) Global (Max)	85,24
		Conv.1 (Sigmoid) Conv.2 (ReLu)	Local (Max) Global (Max)	50,00
		Conv.1 (ReLu) Conv.2 (Sigmoid)	Local (Max) Global (Max)	83,45
		Conv.1 (Sigmoid) Conv.2 (Sigmoid)	Local (Max) Global (Max)	82,15
		Conv.1 (ReLu) Conv.2 (ReLu)	Local (Average) Global (Max)	84,80
		Conv.1 (Sigmoid) Conv.2 (ReLu)	Local (Average) Global (Max)	50,00
		Conv.1 (ReLu) Conv.2 (Sigmoid)	Local (Average) Global (Max)	83,35
		Conv.1 (Sigmoid) Conv.2 (Sigmoid)	Local (Average) Global (Max)	82,01
		Conv.1 (ReLu) Conv.2 (ReLu)	Local (Max) Global (Average)	83,45
		Conv.1 (Sigmoid) Conv.2 (ReLu)	Local (Max) Global (Average)	83,68
		Conv.1 (ReLu) Conv.2 (Sigmoid)	Local (Max) Global (Average)	82,72

		Conv.1 (Sigmoid) Conv.2 (Sigmoid)	Local (Max) Global (Average)	82,07
		Conv.1 (ReLu) Conv.2 (ReLu)	Local (Average) Global (Average)	81,99
		Conv.1 (Sigmoid) Conv.2 (ReLu)	Local (Average) Global (Average)	82,16
		Conv.1 (ReLu) Conv.2 (Sigmoid)	Local (Average) Global (Average)	82,16
		Conv.1 (Sigmoid) Conv.2 (Sigmoid)	Local (Average) Global (Average)	82,24

3.2 Analisis Hasil Eksperimen

Berdasarkan tabel di atas, dapat disimpulkan model struktur pertama (1 *layer*) lebih baik daripada model struktur kedua (2 *layer*) karena tingkat akurasinya lebih tinggi pada uji coba model struktur pertama.

3.3 Kesimpulan Hasil Eksperimen

Dalam perancangan program klasifikasi ulasan film di IMDb menggunakan metode *deep learning* CNN, kami melakukan uji dua model struktur yaitu model struktur dengan 1 *layer* dan model struktur dengan 2 *layer*. Setelah melalui beberapa eksperimen, tingkat akurasi ditemukan pada model struktur pertama ketika mengaktivasi *convolution layer* 1 ReLU dan *pooling* Global (Max).

DAFTAR PUSTAKA

https://docplayer.info/66643898-Klasifikasi-sentimen-review-film-menggunakan-algoritma-support-vector-machine.html

https://www.kaskus.co.id/thread/534b3424148b46c6118b45e9/all-about-imdb-internet-movie-data-base/

https://medium.com/nodeflux/mengenal-convolutional-neural-network-8bd207ad4a8d

https://medium.com/@nadhifasofia/1-convolutional-neural-network-convolutional-neural-network-merupakan-salah-satu-metode-machine-28189e17335b

https://rifqifai.com/implementasi-metode-convolutional-neural-network-untuk-klasifikasi-teks/

https://www.tensorflow.org/tutorials/keras/text_classification_with_hub#download_the_imdb_dataset

LAMPIRAN

Pada direktori yang sama dengan laporan ini, terlampir satu *folder* berisi *code* yang telah diberi *comment* dan dataset yang digunakan.