BAB V
PENUTUP

5.1 Kesimpulan

Bedasarkan hasil dan pembahasan yang dilakukan dari penelitian ini. Beberapa kesimpulan dapat diambil dari penelitian *Penerapan Recurrent Neural Network Untuk Sistem Peringkasan Dan Penerjemahan Teks Berita Bahasa Inggris Ke Bahasa Indonesia*:

1. Model peringkasan dan penerjemahan dibangun menggunakan arsitektur *Encoder-Decoder* dengan menerapkan LSTM (*Long Short Term Memory*). Model peringkasan dibuat menggunakan dataset Gigaword dengan memakai 20% dari dataset atau sejumlah 760.791 data. Model penerjemahan dibangun menggunakan dataset CCMatrix dengan memakai 800.000 data. Kedua dataset dipisah dengan pembagian data 80% untuk latih dan 20% untuk uji validasi. Penelitian menggunakan banyak data tapi tidak dapat menggunakan semua data yang ada karena keterbatasan dari GPU dan memori Google Colaboratory.
2. Hasil model peringkasan mencapai angka *accuracy* sebesar 85% dengan *accuracy* data validasi memiliki nilai 82%. Peringkasan memiliki nilai *loss* terakhir 0.75. Hasil model penerjemahan mencapai angka *accuracy* 84% dan nilai data validasi 81%. Nilai loss untuk model penerjemahan 0.7. Hasil prediksi peringkasan diuji dengan ROUGE, rata-rata ROUGE-1 dan ROUGE-L memiliki nilai 25% dan ROUGE-2 memiliki nilai sangat buruk dengan nilai 8%. Untuk prediksi penerjemahan, rata rata ROUGE-1 dan ROUGE-L memiliki nilai kurang lebih 45% serta ROUGE-2 dengan nilai 24%. Kedua model mengeluarkan nilai bagus saat melakukan pelatihan tapi, saat diuji dengan ROUGE, hasil prediksi mengeluarkan nilai yang kurang baik.
3. Model peringkasan dan penerjemahan digabung menjadi satu *pipeline* melalui API. API dibuat menggunakan *framework* Flask pakai bahasa pemograman Python. API menerima input JSON

dengan teks sebagai tipe input dan mengembalikan JSON dengan hasil peringkasan dan penerjemahan. API ini berbentuk API lokal yang dapat diakses melalui Postman. API menggunakan kedua model yang telah dibuat dan melakukan *inference* melalui input yang diterima lewat JSON.

5.2 Saran

Berikut beberapa saran untuk penelitian selanjutnya:

1. Melakukan lebih banyak *preprocess* untuk meningkatkan hasil nilai *accuracy* saat pelatihan data. Memperluas proses *preprocess* dengan menggunakan *embedding* yang sudah dilatih (seperti Word2Vec) dengan banyak data akan membuat kata keluaran prediksi lebih natural dan sesuai dengan konteks
2. Memakai lebih banyak data agar mendapatkan model dengan ilmu kosakata yang jauh lebih luas dan dapat melakukan prediksi dengan kata yang lebih langka. Jumlah data yang digunakan pada penelitian ini terbatas akibat keterbatasan GPU jadi tidak dapat menggunakan keseluruhan dari dua dataset tersebut.
3. Melakukan lebih banyak eksperimen dengan hyperparameter berbeda untuk mendapatkan model yang lebih optimal. Hyperparameter seperti *epoch* dan besra *batch* dapat diubah untuk mencari konfigurasi model yang menghasilkan metric paling bagus.
4. Menggunakan API di sebuah web untuk menguji fungsionalitas dari API. API dapat di-*deploy* ke sebuah web dan dapat diuji kecepatan dari *inference* untuk model ini.