1. Pendahuluan

Klasifikasi teks merupakan salah satu tugas penting dalam pemrosesan bahasa alami (Natural Language Processing/NLP) karena memungkinkan sistem untuk memahami opini, topik, atau emosi dari sebuah teks. Pada tugas ini, saya membangun model klasifikasi teks menggunakan arsitektur RNN (khususnya LSTM) untuk membedakan ulasan permainan Steam menjadi dua kelas, yaitu positif dan negatif.

2. Dataset

Dataset yang digunakan berasal dari Platform kaggle yaitu: **Steam Review&Games Dataset**

Dataset ini berisi ulasan (review) pengguna terhadap gim di platform Steam, dengan label sentimen "Positive" dan "Negative".

Saya memilih dataset ini karena berasal dari ulasan pengguna nyata di platform Steam, sehingga mewakili opini asli dan alami tanpa proses penyuntingan. Dataset ini juga relevan secara praktis karena dapat digunakan untuk membangun sistem rekomendasi, analisis kepuasan pengguna, atau pemantauan opini komunitas gim. Selain itu, panjang ulasan yang bervariasi menciptakan tantangan yang tepat untuk dieksplorasi menggunakan model berbasis RNN seperti LSTM.

3. Implementasi Model

3.1 Arsitektur RNN

Model yang saya gunakan terdiri dari beberapa lapisan. Pertama, saya menggunakan Embedding Layer yang berfungsi untuk mengubah setiap kata dalam teks menjadi representasi vektor berdimensi tetap. Lapisan ini membantu menangkap makna semantik dari kata-kata tersebut. Selanjutnya, digunakan LSTM (Long Short-Term Memory) dengan 64 unit untuk menangani hubungan temporal dan urutan kata dalam teks. LSTM dipilih karena kemampuannya dalam mengatasi masalah vanishing gradient pada RNN standar dan efektif untuk memahami konteks dalam kalimat panjang. Dropout sebesar 0,5 ditambahkan untuk mengurangi risiko overfitting, dan diakhiri dengan Dense layer beraktivasi sigmoid untuk menghasilkan prediksi biner (positif atau negatif).

3.2 Pra-pemrosesan

Langkah pra-pemrosesan dimulai dengan memuat data dari file CSV. Setelah itu, teks dari ulasan dikonversi menjadi token menggunakan Tokenizer dari Keras, yang membatasi jumlah kata unik maksimum. Kata-kata yang tidak dikenal digantikan dengan token khusus <OOV>. Setelah tokenisasi, setiap ulasan dikonversi menjadi urutan angka berdasarkan indeks kata. Kemudian, dilakukan padding untuk menyamakan panjang setiap input agar sesuai dengan input model. Label sentimen juga dikonversi menjadi bilangan 0 dan 1 agar sesuai dengan format klasifikasi biner.

3.3 Pengaturan Eksperimen

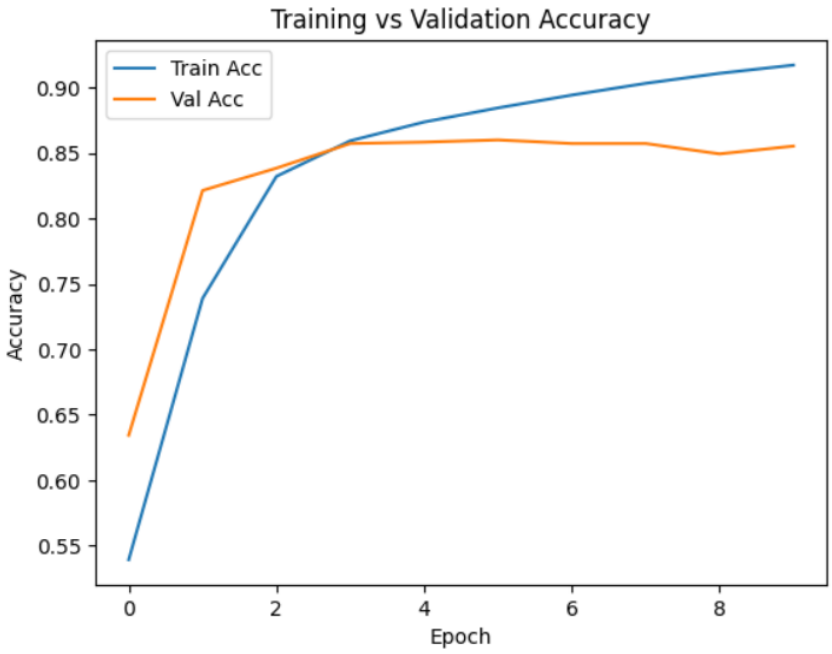
Pengaturan eksperimen yang digunakan adalah 10 epoch, ukuran batch sebesar 32, optimizer Adam, dan fungsi loss Binary Crossentropy. Selain itu, saya menyisihkan sebagian data latih (20%) untuk validasi selama proses pelatihan. Hal ini bertujuan untuk memantau performa model dan mencegah overfitting.

3.4 Log Eksperimen

Percobaan	Model	Dropout	Optimizer	Val Accuracy	Catatan
#1	LSTM(64)	0	Adam	83,20%	Mulai overfitting di epoch ke-4
#2	LSTM(64)	0,5	Adam	85,50%	Lebih stabil, val_loss menurun

4. Evaluasi Hasil

Model dievaluasi menggunakan akurasi dan loss dari fungsi model.evaluate() terhadap data uji. Hasil evaluasi menunjukkan bahwa model mampu membedakan ulasan positif dan negatif dengan cukup baik. Selain itu, saya juga menggunakan confusion matrix untuk melihat distribusi klasifikasi benar dan salah, serta classification report dari sklearn untuk metrik tambahan seperti precision, recall, dan F1-score.



5. Refleksi Pribadi

Selama mengerjakan tugas ini, tantangan utama yang saya hadapi adalah mengelola overfitting pada model LSTM. Pada percobaan awal tanpa dropout, model dengan cepat mencapai akurasi tinggi pada data latih, tetapi performa validasi memburuk setelah beberapa epoch. Hal ini mengindikasikan bahwa model mengingat data latih terlalu baik dan tidak mampu melakukan generalisasi terhadap data baru.

Untuk mengatasi masalah tersebut, saya menambahkan lapisan Dropout dengan rate 0,5 yang terbukti cukup efektif dalam menstabilkan akurasi validasi. Selain itu, saya juga mengeksplorasi pemangkasan jumlah epoch agar model tidak dilatih terlalu lama dan menghindari overfitting.

Pelajaran penting dari tugas ini adalah pentingnya eksperimen bertahap, pengujian konfigurasi, serta tidak bergantung hanya pada satu metrik evaluasi. Saya juga belajar bagaimana membaca hasil evaluasi dan log pelatihan untuk menyimpulkan performa model secara menyeluruh. Menggunakan bantuan ChatGPT dalam proses ini sangat membantu mempercepat pemahaman arsitektur LSTM, namun tetap saya verifikasi dengan eksperimen langsung dan observasi manual terhadap hasilnya.

6. Kesimpulan dan Saran

Model LSTM yang dibangun pada tugas ini berhasil mencapai akurasi validasi sekitar 85% dengan performa yang stabil setelah penambahan dropout. Dataset ulasan Steam terbukti cocok untuk tugas klasifikasi sentimen karena mengandung opini nyata pengguna yang ekspresif dan variatif. Ke depan, saya menyarankan untuk mencoba model lain seperti BiLSTM atau GRU, menambah jumlah data untuk meningkatkan generalisasi, serta mengeksplorasi penggunaan word embeddings pra-latih seperti GloVe atau FastText agar model dapat memahami konteks kata secara lebih mendalam.