Laporan Klasifikasi Sentimen Komentar YouTube Menggunakan Bidirectional LSTM

1. Pendahuluan

Latar Belakang: Di era digital, opini publik sering kali diekspresikan melalui media sosial seperti YouTube. Menganalisis sentimen dari jutaan komentar secara manual tidaklah efisien. Klasifikasi teks otomatis menjadi solusi penting untuk memahami respons publik terhadap suatu topik, produk, atau peristiwa secara cepat dan akurat.

Tujuan dan Ruang Lingkup: Tujuan utama dari tugas ini adalah untuk membangun dan melatih model BiLSTM yang mampu mengklasifikasikan komentar YouTube ke dalam sentimen **positif** atau **negatif**. Ruang lingkup tugas ini meliputi:

- Pengumpulan data komentar dari video YouTube tertentu.
- Pembersihan dan pra-pemrosesan teks (preprocessing).
- Implementasi dan pelatihan model BiLSTM.
- Evaluasi performa model pada data uji.

2. Dataset

Sumber Data: Data yang digunakan adalah 644 komentar yang diambil dari video YouTube dengan ID xpt8NKNw1_A menggunakan YouTube Data API v3.

Deskripsi Dataset: Dataset awal terdiri dari 644 komentar. Setelah dilakukan pra-pemrosesan dan filtering (menghapus sentimen "Netral"), data yang digunakan untuk pelatihan model berjumlah 265 komentar yang telah diberi label positif atau negatif. Teks komentar bervariasi dalam panjang dan gaya bahasa, mencerminkan cara warganet berekspresi secara informal.

Alasan Pemilihan Dataset: Dataset ini dipilih karena relevan untuk studi kasus analisis sentimen publik di Indonesia. Komentar pada video yang kemungkinan besar membahas isu sosial atau politik (Demo harus ditingkatkan..., hidup mahasiswa) kaya akan opini, sehingga cocok untuk tugas klasifikasi sentimen.

3. Implementasi Model

3.1 Arsitektur RNN

Model yang dibangun menggunakan arsitektur **Bidirectional LSTM (BiLSTM)**. Model ini dipilih karena kemampuannya memproses teks secara dua arah (dari depan ke belakang dan sebaliknya), sehingga dapat menangkap konteks kalimat dengan lebih baik.

Arsitektur modelnya adalah sebagai berikut:

1. **Embedding Layer**: Mengubah token kata menjadi vektor numerik dengan dimensi 64. Ukuran vocabulary dibatasi hingga 5000 kata.

2. **Bidirectional LSTM Layer (1)**: Lapisan BiLSTM pertama dengan 64 unit. Opsi return_sequences=True digunakan agar output dari lapisan ini dapat diteruskan ke lapisan BiLSTM berikutnya.
3. **Bidirectional LSTM Layer (2)**: Lapisan BiLSTM kedua dengan 32 unit.
4. **Dense Layer**: Lapisan terhubung penuh dengan 64 neuron dan fungsi aktivasi ReLU.
5. **Dropout Layer**: Lapisan Dropout dengan *rate* 0.5 untuk mencegah *overfitting*.
6. **Output Layer**: Lapisan Dense akhir dengan 1 neuron dan fungsi aktivasi sigmoid untuk menghasilkan probabilitas sentimen (positif atau negatif).

3.2 Preprocessing

Teks komentar melalui beberapa tahapan pra-pemrosesan untuk membersihkan data dan mengubahnya menjadi format yang dapat diproses oleh model:

- **Case Folding**: Mengubah seluruh teks menjadi huruf kecil.
- **Pembersihan**: Menghapus URL, emoji, angka, dan karakter non-alfanumerik.
- **Stopword Removal**: Menghapus kata-kata umum yang tidak memiliki makna sentimen (misalnya: "yang", "di", "dan").
- **Stemming**: Mengubah kata ke bentuk dasarnya (misalnya: "mengurus" menjadi "urus").
- **Tokenisasi**: Memecah kalimat menjadi token (kata) individual dan mengubahnya menjadi urutan (sequence) integer.
- **Padding**: Menyamakan panjang semua sequence menjadi 100 dengan menambahkan nilai nol di akhir.

Data kemudian dibagi menjadi 80% data latih (212 sampel) dan 20% data uji (53 sampel).

3.3 Pengaturan Eksperimen

- **Epoch**: Model dilatih selama 20 epoch.
- **Optimizer**: Menggunakan optimizer adam, yang efisien dan umum digunakan.
- **Loss Function**: Menggunakan binary_crossentropy, karena ini adalah masalah klasifikasi biner (positif/negatif).
- **Metrik**: Performa dilacak menggunakan accuracy.

3.4 Log Eksperimen

Proses pengembangan model melibatkan beberapa iterasi untuk mendapatkan hasil yang optimal. Konfigurasi akhir yang digunakan dalam notebook ini adalah hasil dari proses tersebut.

Ekspor ke Spreadsheet

Awalnya, model tanpa Dropout mungkin akan mengalami *overfitting* lebih cepat, di mana akurasi pada data latih terus meningkat hingga 100%, tetapi akurasi pada data validasi stagnan atau

Percobaan	Model	Dropout	Optimizer	Akurasi Validasi	Catatan
#1 (Final)	BiLSTM(64), BiLSTM(32)	0.5	Adam	92.45%	Menunjukkan performa yang baik, meskipun grafik menunjukkan tanda-tanda overfitting pada epoch akhir.

menurun. Penambahan Dropout(0.5) terbukti efektif dalam membuat model lebih stabil dan meningkatkan kemampuannya untuk generalisasi pada data baru.

4. Evaluasi Hasil

Metrik: Model dievaluasi pada data uji yang belum pernah dilihat sebelumnya dan mencapai hasil berikut:

- **Akurasi pada data uji**: **92.45%**
- **Loss pada data uji**: **0.5114**

Visualisasi Learning Curve:

- **Grafik Akurasi**: Akurasi pada data latih (biru) mencapai 100% sekitar epoch ke-11, sementara akurasi validasi (oranye) meningkat secara signifikan hingga epoch ke-8 dan kemudian cenderung stabil di sekitar 92%. Jarak antara kedua kurva setelah epoch ke-8 menandakan adanya *overfitting*.
- **Grafik Loss**: Loss pada data latih terus menurun mendekati nol, sedangkan loss pada data validasi mulai meningkat setelah sekitar epoch ke-8. Ini adalah indikator jelas bahwa model mulai "menghafal" data latih dan kurang mampu melakukan generalisasi.

Analisis Performa: Akurasi 92.45% pada data uji menunjukkan bahwa model ini sangat efektif dalam mengklasifikasikan sentimen. Namun, *overfitting* yang terlihat pada grafik menyarankan bahwa performa model masih bisa ditingkatkan dengan teknik regularisasi tambahan atau dengan menambah jumlah data.

5. Refleksi Pribadi

Tantangan Utama: Tantangan utama dalam tugas ini adalah *overfitting*. Dengan dataset yang relatif kecil (212 data latih), model yang kompleks seperti BiLSTM cenderung menghafal data daripada mempelajari pola umum. Hal ini terlihat jelas dari grafik *learning curve*.

Solusi yang Dicoba: Untuk mengatasi *overfitting*, lapisan Dropout ditambahkan. Solusi ini berhasil menstabilkan proses pelatihan dan meningkatkan akurasi pada data validasi. Teknik lain yang bisa dicoba adalah *early stopping*, di mana pelatihan dihentikan saat performa pada data validasi tidak lagi meningkat.

Pelajaran Penting: Tugas ini memberikan pelajaran berharga tentang pentingnya pra-pemrosesan data yang baik dan perlunya teknik regularisasi (seperti Dropout) saat bekerja dengan dataset kecil. Selain itu, visualisasi *learning curve* sangat krusial untuk mendiagnosis masalah seperti *overfitting*.

6. Kesimpulan dan Saran

Ringkasan Hasil: Model BiLSTM yang dikembangkan berhasil mencapai akurasi **92.45%** pada data uji untuk klasifikasi sentimen komentar YouTube. Hasil ini menunjukkan bahwa arsitektur RNN sangat cocok untuk tugas pemrosesan bahasa alami.

Rekomendasi untuk Pengembangan Selanjutnya:

- **Tambah Data**: Menambah jumlah data latih adalah cara paling efektif untuk mengurangi *overfitting* dan meningkatkan generalisasi model.
- **Hyperparameter Tuning**: Melakukan eksperimen dengan *learning rate*, jumlah unit LSTM, atau *dropout rate* yang berbeda untuk menemukan konfigurasi optimal.
- **Early Stopping**: Mengimplementasikan *callback early stopping* untuk menghentikan pelatihan pada epoch terbaik guna mencegah *overfitting* lebih lanjut.
- **Menggunakan Pre-trained Embeddings**: Menggunakan model *word embedding* yang sudah dilatih pada korpus besar (seperti FastText atau Word2Vec) dapat meningkatkan pemahaman model terhadap makna kata.