LAPORAN AKHIR TUGAS PBA

PERBANDINGAN METODE FASTTEXT, TRANSFORMER, DAN LSTM UNTUK KLASIFIKASI TEKS MENGGUNAKAN WORD EMBEDDING GLOVE PADA DATASET AGNEWS

Disusun Oleh :

Hermalina Sintia Putri	121450052
Veni Zahara Kartika	121450075
Syifa Firnanda	121450094
Silvia Azahrani	121450070
Wahyudianto	121450040

INSTITUT TEKNOLOGI SUMATERA

FAKULTAS SAINS

PROGRAM STUDI SAINS DATA

1. Metode FastText

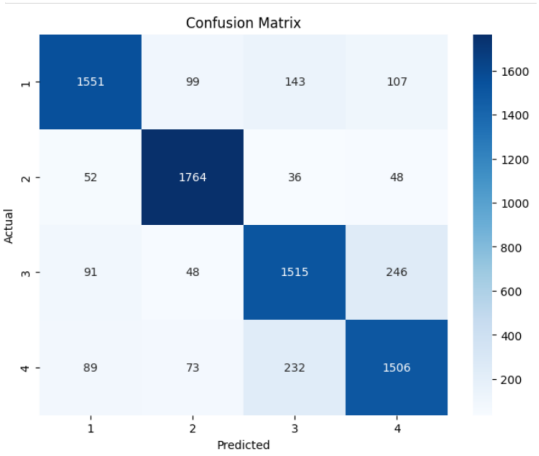
Deskripsi Metode
FastText adalah pendekatan sederhana namun efektif untuk pemodelan teks dengan representasi kata menggunakan Word Embedding dari GloVe. FastText tidak memperhitungkan urutan kata dalam teks karena menggunakan mean pooling, yakni menghitung rata-rata embedding dari semua token (kata) dalam teks untuk mendapatkan representasi vektor keseluruhan. FastText sangat efisien dari sisi komputasi dan cepat untuk dilatih.

Implementasi

- Dataset: Dataset yang digunakan adalah kombinasi title dan description dari AGNews. Dataset dibagi menjadi train dan test.
- Word Embedding: Menggunakan pre-trained GloVe embeddings dengan dimensi 100.
- Arsitektur Model:
    - Embedding Layer: GloVe embeddings digunakan tanpa training ulang (freeze=True).
    - Mean Pooling: Rata-rata embedding dari seluruh kata dalam teks.
    - Fully Connected (FC) Layer: Digunakan untuk memprediksi kelas berdasarkan representasi vektor teks.
- Training:
    - Optimizer: Adam
    - Loss: CrossEntropyLoss
    - Epochs: 10
    - Batch size: 32

Hasil Pengujian

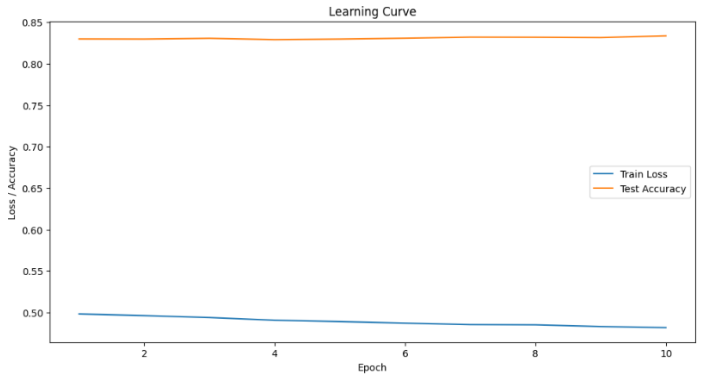
1. Confusion Matrix:



Gambar 1. Convution matriks

    - Label 1: Recall 82%, Precision 87%.
    - Label 2: Recall 93%, Precision 89%.
    - Label 3: Recall 80%, Precision 79%.

    - Label 4: Recall 79%, Precision 79%.
2. Classification Report:
    - Accuracy: 83%
    - Macro average (F1-Score): 83%
    - Weighted average (F1-Score): 83%
3. Learning Curve:



Gambar 2. Learning rate

    - Train Loss berkurang stabil dari 1.03 ke 0.50.
    - Test Accuracy meningkat dan stabil di 82-83%.
4. Waktu dan Sumber Daya:
    - GPU Memory Usage: 222.89 MB digunakan dari total 230.69 MB.
    - Rata-rata waktu per batch:
        - Train: 0.0015 detik
        - Test: 0.0013 detik

Analisis
FastText memiliki keunggulan dari sisi efisiensi komputasi. Akurasi mencapai 83%, yang cukup baik untuk dataset AGNews. Namun, kelemahan utama FastText adalah tidak memperhitungkan urutan kata, sehingga kesulitan menangkap informasi semantik yang kompleks dalam teks.

2. Metode Transformer

Deskripsi Metode
Transformer adalah model attention-based yang digunakan untuk menangkap hubungan antar kata dalam teks secara paralel. Berbeda dengan FastText, Transformer mampu memahami urutan dan konteks antar token menggunakan self-attention mechanism.

Implementasi

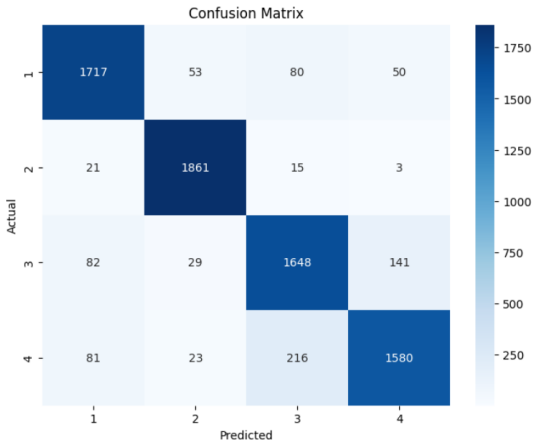
- Dataset: Dataset yang digunakan adalah kombinasi title dan description dari AGNews.
- Word Embedding: Menggunakan GloVe embeddings dengan dimensi 100.
- Arsitektur Model:

- Embedding Layer: Layer embedding dengan GloVe pretrained embeddings (freeze=True).
- Positional Encoding: Ditambahkan untuk memberikan informasi posisi token dalam teks.
- Transformer Encoder:
  - Jumlah Layer: 2
  - Attention Heads: 4
  - Feedforward Hidden Size: 128
- Fully Connected Layer: Klasifikasi ke 4 kelas.

- Training:
  - Optimizer: Adam
  - Loss: CrossEntropyLoss
  - Epochs: 5
  - Batch size: 32

Hasil Pengujian

1. Confusion Matrix:



Gambar 3. Convution matriks transformer

   - Label 2 memiliki performa terbaik dengan recall 98% dan F1-score 96%.
   - Label 3 memiliki F1-score lebih rendah, yaitu 85%, menunjukkan beberapa kesalahan prediksi ke kelas lain.
2. Classification Report:
   - Accuracy: 90%
   - Macro average (F1-Score): 90%
   - Weighted average (F1-Score): 90%
3. Learning Curve:
   - Akurasi validasi meningkat pada Epoch 1-2, namun menurun di Epoch 3 dan naik kembali di Epoch 4-5.
   - Akurasi validasi tertinggi tercapai di Epoch 5: 89.58%.
4. Waktu dan Sumber Daya:
   - GPU Memory Usage: 181.19 MB digunakan dari total 255.85 MB.
   - Rata-rata waktu per batch:
     - Train: 0.0073 detik
     - Test: 0.0017 detik

Analisis
Transformer mampu menangkap urutan kata dan konteks dengan lebih baik dibanding FastText. Akurasi yang dicapai adalah 90%, tertinggi di antara metode lainnya. Namun, Transformer memerlukan waktu training lebih lama dan konsumsi memori yang lebih tinggi.

3. Metode LSTM

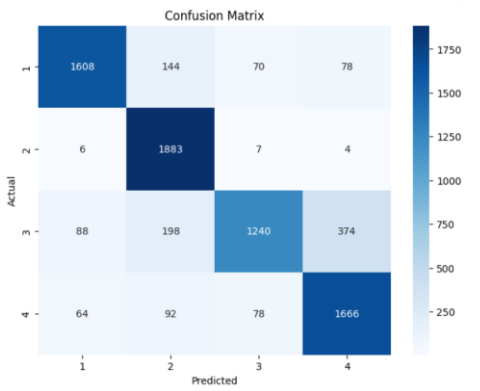
Deskripsi Metode
LSTM (Long Short-Term Memory) adalah model recurrent neural network (RNN) yang dirancang untuk menangani long-term dependencies dalam data sekuensial seperti teks. LSTM bekerja dengan mempertahankan state memori melalui jaringan sel dan gate mechanism.

Implementasi

- Dataset: Dataset yang digunakan adalah kombinasi title dan description dari AGNews.
- Word Embedding: GloVe embeddings dengan dimensi 100.
- Arsitektur Model:
    - Embedding Layer: Menggunakan GloVe pretrained embeddings (freeze=True).
    - LSTM Layer:
        - Hidden size: 128
        - Batch-first: True
    - Fully Connected Layer: Prediksi kelas dari state terakhir LSTM.
    - Dropout: 0.5 untuk regularisasi.
- Training:
    - Optimizer: Adam
    - Loss: CrossEntropyLoss
    - Epochs: 10
    - Batch size: 32

Hasil Pengujian

1. Confusion Matrix:



Gambar 4. Convution matriks LSTM

- Label 2 memiliki performa sangat baik dengan recall 99% dan F1-score 89%.
- Label 3 menunjukkan kelemahan dengan recall 65%, artinya model sering salah memprediksi kelas ini ke Label 4.

2. Classification Report:
   - Accuracy: 84%
   - Macro average (F1-Score): 84%
   - Weighted average (F1-Score): 84%
3. Train Loss dan Akurasi:
   - Train Loss menurun stabil dari 1.38 menjadi 0.45.
   - Test Accuracy meningkat signifikan di awal dan stabil di 84% setelah Epoch 5.
4. Waktu dan Sumber Daya:
   - GPU Memory Usage: 647.03 MB digunakan dari total 796.92 MB.
   - Rata-rata waktu per epoch adalah 14 detik.

Analisis
LSTM mampu menangkap urutan kata lebih baik dibanding FastText. Akurasi yang dicapai adalah 84%, namun model masih memiliki kelemahan pada Label 3 dengan recall yang rendah. LSTM membutuhkan waktu training lebih lama dan memori lebih besar dibanding FastText.

Perbandingan FastText, Transformer, dan LSTM

Pada aspek **dataset**, ketiga metode menunjukkan performa yang berbeda. FastText, meskipun sederhana, dapat bekerja dengan baik pada dataset kecil hingga menengah karena menggunakan mean pooling yang tidak memerlukan pemahaman kompleks dari urutan kata. Namun, metode ini mungkin akan membutuhkan dataset yang lebih besar jika data memiliki banyak variasi semantik. Transformer, sebagai model yang lebih canggih dengan *self-attention mechanism*, memanfaatkan hubungan antar token lebih baik dan cenderung memiliki performa yang lebih stabil seiring peningkatan ukuran dataset. LSTM, meskipun lebih baik dalam menangkap urutan kata dibandingkan FastText, memerlukan dataset yang cukup besar untuk mempelajari pola secara efektif dan menghindari overfitting, terutama untuk kelas yang sulit seperti yang terlihat pada performa kelas 3.

Dari segi **waktu dan sumber daya komputasi**, FastText adalah yang paling efisien, baik dalam hal waktu training maupun penggunaan memori GPU, dengan rata-rata waktu per batch sekitar **0.0015 detik** dan konsumsi memori yang rendah (**222.89 MB**). Transformer memerlukan waktu lebih lama (**0.0073 detik per batch**) dan konsumsi memori yang moderat (**181.19 MB**) karena kompleksitas *self-attention mechanism* dan posisional encoding. LSTM berada di antara keduanya dalam hal waktu (**14 detik per epoch**) namun memiliki konsumsi memori tertinggi (**647.03 MB**), karena LSTM mempertahankan state memori yang memerlukan komputasi tambahan.

Dalam hal **generalisasi**, Transformer memiliki performa terbaik dengan akurasi **90%** karena kemampuannya menangkap konteks kata secara menyeluruh. LSTM mampu memahami urutan kata dengan baik dan mencapai akurasi **84%**, namun menunjukkan kelemahan pada kelas yang

kompleks atau mirip (kelas 3). Sementara itu, FastText memiliki generalisasi yang cukup baik dengan akurasi **83%**, tetapi keterbatasannya dalam memahami urutan kata membuatnya kurang akurat untuk data dengan konteks yang kompleks. Secara keseluruhan, Transformer lebih unggul dalam generalisasi, terutama ketika dataset memadai, sedangkan FastText adalah solusi ringan yang efektif untuk dataset sederhana.