Transformers

Production-ready Transformer implementation menggunakan pretrained BERT dan Hugging Face Transformers library. Spam classification dan question answering tasks.[1]

Transformer Components: Advanced Details

Advanced Embeddings:

```
class TransformerEmbedding(tf.keras.layers.Layer):
    def __init__(self, vocab_size, d_model):
        super().__init__()
        self.token_emb = Embedding(vocab_size, d_model)
        self.pos_emb = PositionalEncoding(d_model)

    def call(self, inputs):
        seq_len = tf.shape(inputs)[^1]
        positions = tf.range(start=0, limit=seq_len, delta=1)
        pos_emb = self.pos_emb(positions)
        token_emb = self.token_emb(inputs)
        return token_emb + pos_emb
```

Residual Connections + Layer Normalization:

```
def transformer_block(x, num_heads, d_model, dff, rate=0.1):
    attn_output = MultiHeadAttention(num_heads, d_model)(x, x, x)
    attn_output = Dropout(rate)(attn_output)
    out1 = LayerNormalization(epsilon=1e-6)(x + attn_output)

    ffn_output = Dense(dff, activation='relu')(out1)
    ffn_output = Dense(d_model)(ffn_output)
    ffn_output = Dropout(rate)(ffn_output)
    out2 = LayerNormalization(epsilon=1e-6)(out1 + ffn_output)
    return out2
```

BERT untuk Spam Classification

TFHub BERT Loading:

```
def create_bert_classifier():
    bert_preprocess = hub.KerasLayer("https://tfhub.dev/tensorflow/bert_en_uncased_prepro
    bert_encoder = hub.KerasLayer("https://tfhub.dev/tensorflow/bert_en_uncased_L-12_H-76

    inputs = Input(shape=(), dtype=tf.string)
    encoder_inputs = bert_preprocess(inputs)
    bert_output = bert_encoder(encoder_inputs)
    pooled_output = bert_output["pooled_output"]

    x = Dropout(0.1)(pooled_output)
    outputs = Dense(1, activation='sigmoid')(x)

    model = Model(inputs, outputs)
    return model
```

Fine-tuning Strategy:

```
model.compile(
    optimizer=tf.keras.optimizers.Adam(learning_rate=2e-5),
    loss='binary_crossentropy',
    metrics=['accuracy']
)

# Freeze BERT weights awalnya, kemudian fine-tune
bert_encoder.trainable = True
```

Question Answering dengan Hugging Face Transformers

DistilBERT untuk SQuAD:

```
from transformers import TFAutoModelForQuestionAnswering, AutoTokenizer

tokenizer = AutoTokenizer.from_pretrained("distilbert-base-cased-distilled-squad")
model = TFAutoModelForQuestionAnswering.from_pretrained("distilbert-base-cased-distilled-

def extract_answer(question, context):
    inputs = tokenizer(question, context, return_tensors="tf", max_length=384, truncation

    outputs = model(**inputs)
    start_logits = outputs.start_logits
    end_logits = outputs.end_logits

    start_idx = tf.argmax(start_logits, axis=1)
    end_idx = tf.argmax(end_logits, axis=1)

    answer = tokenizer.decode(inputs.input_ids[^0][start_idx:end_idx])
    return answer
```

Training QA Model

```
def qa_loss(y_true, y_pred):
    start_pred, end_pred = y_pred
    start_true, end_true = y_true[:, :, 0], y_true[:, :, 1]
    start_loss = tf.keras.losses.sparse_categorical_crossentropy(start_true, start_pred)
    end_loss = tf.keras.losses.sparse_categorical_crossentropy(end_true, end_pred)
    return (start_loss + end_loss) / 2
```

Kesimpulan

Pretrained Transformers BERT, DistilBERT) dengan fine-tuning adalah gold standard untuk production NLP tasks. [1]