Transformer Challenge

Enrique Gómez Cruz

March 21, 2025

Abstract

This report presents the results of training and evaluating four sequence-to-sequence transformer models for neural machine translation. The models differ in their configurations. Two models utilize components from the Keras Hub library, with variations in their loss functions. One model is built entirely from scratch. Another model uses pretrained GloVe embeddings for the encoder's embedding layers. All models were trained with the same hyperparameters: 100 epochs, batch size of 64, maximum sequence length of 20 tokens, vocabulary size of 20,000 tokens (10,000 for the pretrained model), embedding dimension of 256 (100 for GloVe), latent dimension of 2048, and 8 attention heads. Model performance was evaluated using ROUGE and BLEU scores on 30 test sentence pairs, where the scratch and pretrained models significantly outperformed the Keras Hub models. The pretrained model delivered comparable translation performance to the scratch model but had a smaller size (35 MB vs. 191 MB). The report provides detailed comparisons of accuracy, loss curves, and example translations across models.

1 Introduction

Four sequence-to-sequence transformer models are trained. Two are built using components from the `keras_hub` library, differing only in their loss functions. Another is trained from scratch, while the final model uses pretrained GloVe embeddings in its embedding layers.

All models are trained using the same parameters, except where explicitly stated otherwise:

- 100 training epochs.
- Batches of size 64.
- A max sequence length of 20 tokens.
- Max vocabulary size of 20,000 tokens.
- An embedded dimension of 256.
- A latent dimension of 2048.
- 8 attention heads.

The models are trained in Google Colab and saved to disk with a `ModelCheckpoint` callback. Similarly, its training history is saved to disk using the `CSVLogger` callback function.

```
# csv logger callback
csv_logger = CSVLogger("transformer.training.log")

# model checkpoint callback
model_checkpoint = ModelCheckpoint(
    filepath="transformer.checkpoint.keras",
    monitor="val_accuracy",
    mode="max",
    save_best_only=True,
)

transformer.fit(
    train_ds,
    epochs=epochs,
    validation_data=val_ds,
    verbose=2,
    callbacks=[csv_logger, model_checkpoint],
)
```

Models can be loaded from disk with the `load_model` function from Keras. A dictionary (`custom_objects`) describing the objects to deserialize is needed.

```
transformer = keras.models.load_model(
    "transformer.checkpoint.keras",
    custom_objects={
        "encoder_inputs": encoder_inputs,
        "decoder_inputs": decoder_inputs,
        "TransformerEncoder": TransformerEncoder,
        "TransformerDecoder": TransformerDecoder,
        "PositionalEmbedding": PositionalEmbedding,
    },
)
```

The following sections provide a detailed description of the models, arranged in order of increasing performance. A performance summary of the four models is shown in table 1.

Table 1: ROUGE and BLEU metric score of the transformer models. The metric is calculated using 30 test pair sentences. The Rouge score displayed is the 'f1 score' that summarizes recall and precision scores.

Model	Rouge-1	Rouge-2	Bleu-1	Bleu-2
Keras Hub v1	0.195	0.0	0.380	0.276
Keras Hub v2	0.226	0.003	0.400	0.258
Scratch	0.701	0.429	0.667	0.436
Pretrained	0.718	0.418	0.625	0.408

2 Keras Hub model (v1 & v2)

This model is based on the tutorial of Abheesht Sharma and uses predefined components from the `keras_hub` library.

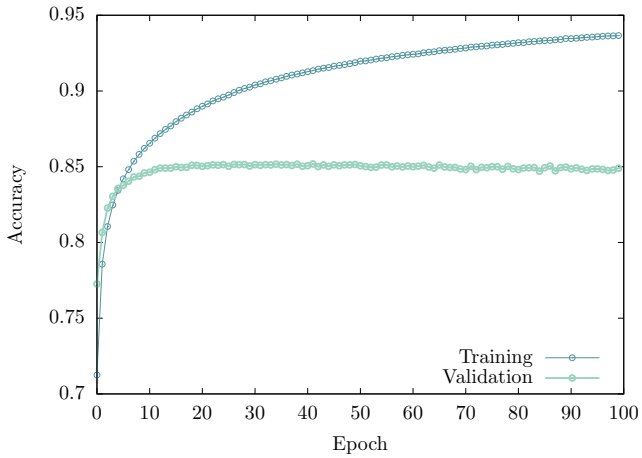

Figure 1: Validation and training accuracy for the Keras Hub model.

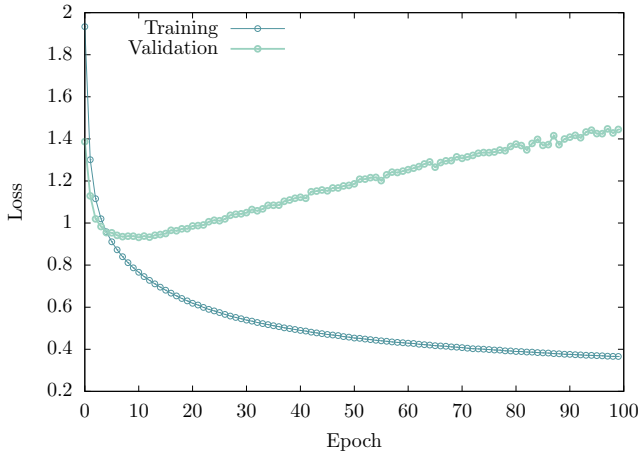

Figure 2: Validation and training loss for the Keras Hub model.

The model achieves a surprisingly high validation accuracy even though it does poorly at translating sentences, as seen from the following examples:

Keras Hub model translation examples

```
The children love listening to stories.
[start] algo cómo hacer aprender existen parecer japonés ya
quién seis de [end]

The buildings are small in comparison to the skyscrapers
in New York.
[start] algo estúpido están entienda empieza parecer usted
pagaron palabras y héroe parecer próxima hacer aprender
entienda honesto malo habla [end]

He suggested to me that we go to the beach.
[start] algo para mire estado para tímido un está se ya y
hijo seguro existen somos japonés de ya quién [end]

Tom thinks women in America wear too much perfume.
[start] nada misma estás prefiero ya ventanas y hijo robar
salvado quedan robaron seguridad se [end]

I've spent all the money.
[start] puedes tienes adónde y parecer aprender venido
empezar se [end]

He thought someone had put poison in his soup.
[start] y herido japonés talla pidió dos yo sabemos cliente
claro pero número se [end]

I knew you'd be in here.
[start] y héroe sabemos entienda razón prisa ya gustaría
arroz y cuánto borracho se [end]

I'm the only one who had to do that.
[start] algo ese entienda empieza estado robar estará podría
estado de pero las hacen parado se [end]

Tom claimed to be the son of a rich man.
[start] algo al tinto ve clave pagaron palabras río perdida
cantó para cómo estás nunca aquí sitio finales cliente
hablando [end]

Tom is well aware of the problem.
[start] para estás quién reír atención para existen resulta
diría se [end]
```

The high accuracy is achieved due to the fact that the loss function is not ignoring the class corresponding to the empty token ('') in the vocabulary. After fixing that issue the curves look like the ones in figure 3 and figure 4.

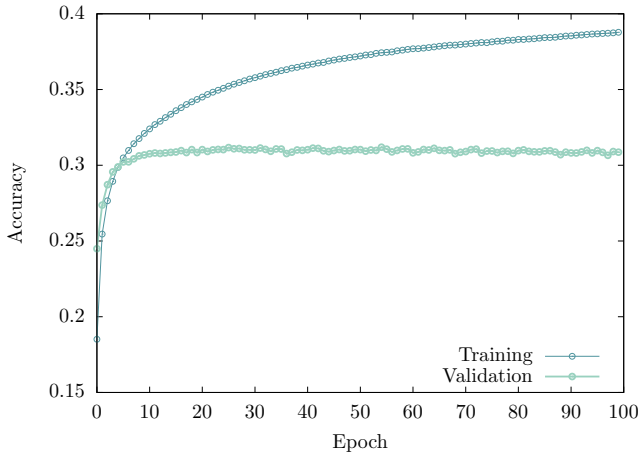

Figure 3: Validation and training accuracy for the Keras Hub model after ignoring the empty token class in the loss function.

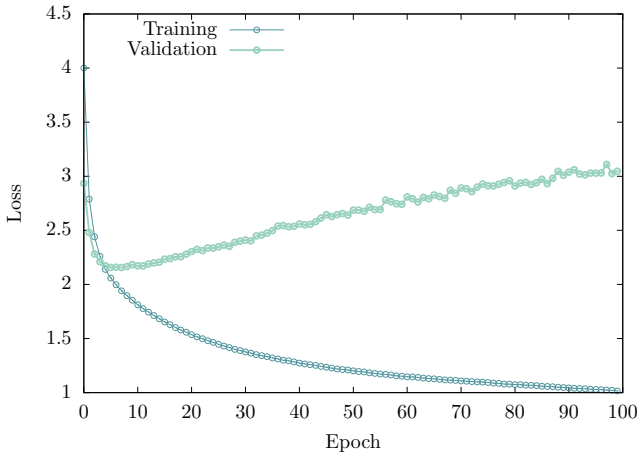

Figure 4: Validation and training loss for the Keras Hub model after ignoring the empty token class in the loss function.

3 Scratch model

This model is based on the tutorial of François Chollet and builds the transformer architecture from scratch.

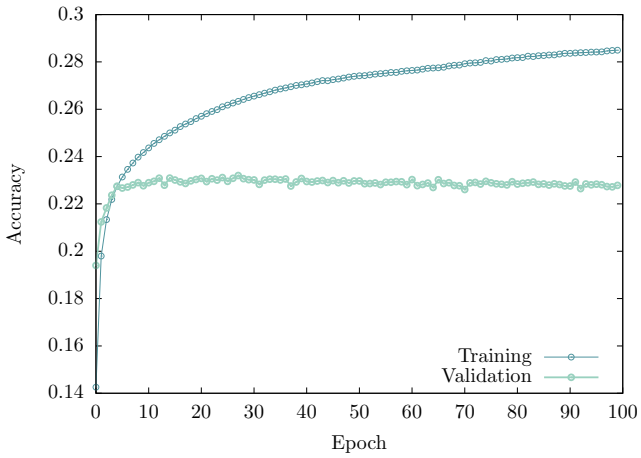

Figure 5: Validation and training accuracy for the scratch model.

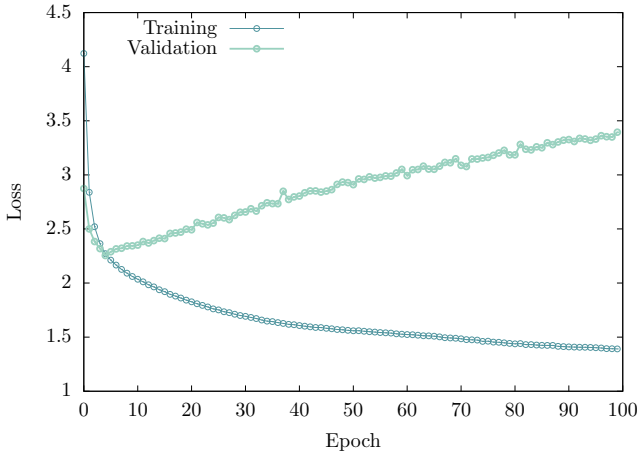

Figure 6: Validation and training loss for the scratch model.

Although the accuracy of this model is lower than the previous one, it does a much better job at translating, as seen in the following examples:

Scratch model translation examples

```
The children love listening to stories.
[start] los niños quieren oír las historias [end]

The buildings are small in comparison to the skyscrapers
in New York.
[start] los [UNK] son nuevas [UNK] a los nuevas varios
nuevas en nueva york [end]

He suggested to me that we go to the beach.
[start] me [UNK] que vaya a la playa [end]

Tom thinks women in America wear too much perfume.
[start] tom cree que las mujeres américa en un [UNK]
demasiado ser feo [end]

I've spent all the money.
[start] he pasado todo el dinero [end]

He thought someone had put poison in his soup.
[start] pensó que alguien tenía una sopa de veneno [end]

I knew you'd be in here.
[start] sabía que estarías aquí [end]

I'm the only one who had to do that.
[start] soy tú el único que tuvo que hacer eso [end]

Tom claimed to be the son of a rich man.
[start] tom [UNK] ser rica un hombre de un hombre rica [end]

Tom is well aware of the problem.
[start] tom tiene muy capaz de resolver el problema [end]
```

4 Pretrained model

This model is trained from scratch with the same layers as the previous one, except for the weights of the embedding layer used for the encoding inputs (english sentences) which are GloVe pretrained weights. This model uses an embedded dimension of 100 since it is the dimension of the GloVe embeddings used. The max vocabulary size had to be reduced to 10,000 tokens since Google Colab crashed with an error that I couldn't fix, even though locally the model was trainable (although very slowly).

The first step is to load the weights of the GloVe embeddings:

```python
path_to_glove_file = "../glove.6b/glove.6B.100d.txt"

embeddings_index = {}
with open(path_to_glove_file) as f:
    for line in f:
        word, coefs = line.split(maxsplit=1)
        coefs = np.fromstring(coefs, "f", sep=" ")
        embeddings_index[word] = coefs

voc = fre_vectorization.get_vocabulary()
vocab_size = len(voc) + 2
```

```
hits = 0
misses = 0

# Prepare embedding matrix
word_index = dict(zip(voc, range(len(voc))))
embedding_matrix = np.zeros((vocab_size, embed_dim))
for word, i in word_index.items():
    embedding_vector = embeddings_index.get(word)
    if embedding_vector is not None:
        # Words not found in embedding index will be
        ↪ all-zeros.
        # This includes the representation for "padding"
        ↪ and "OOV"
        embedding_matrix[i] = embedding_vector
        hits += 1
    else:
        misses += 1
```

The `PositionalEmbedding` class is modified to accept the `trainable` parameter. When the layer is set to non-trainable its weights come from the embedding matrix previously loaded.

```
class PositionalEmbedding(layers.Layer):
    def __init__(self, sequence_length, vocab_size,
    ↪ embed_dim, **kwargs):
        super().__init__(**kwargs)
        self.token_embeddings = layers.Embedding(
            input_dim=vocab_size, output_dim=embed_dim
        )
        self.position_embeddings = layers.Embedding(
            input_dim=sequence_length,
            ↪ output_dim=embed_dim
        )
        self.sequence_length = sequence_length
        self.vocab_size = vocab_size
        self.embed_dim = embed_dim

        if not self.trainable:
            self.token_embeddings.trainable = False
            self.token_embeddings.build((1,))
            self.token_embeddings.set_weights(
            ↪ [embedding_matrix])
```

This and the scratch model have almost the same performance, but they differ vastly on size: 35 MB vs. 191 MB.

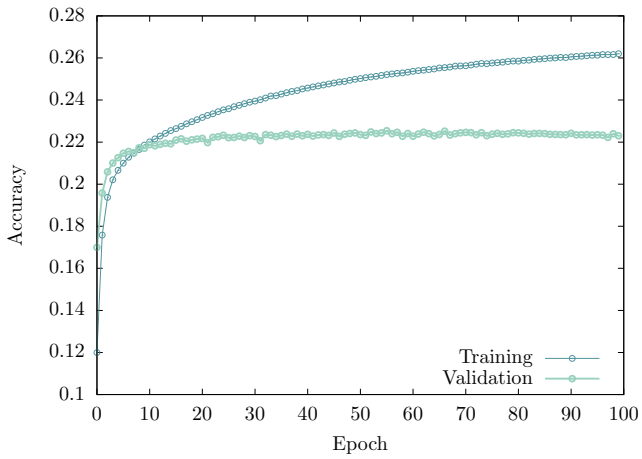

Figure 7: Validation and training accuracy for the pretrained model with GloVe embeddings.

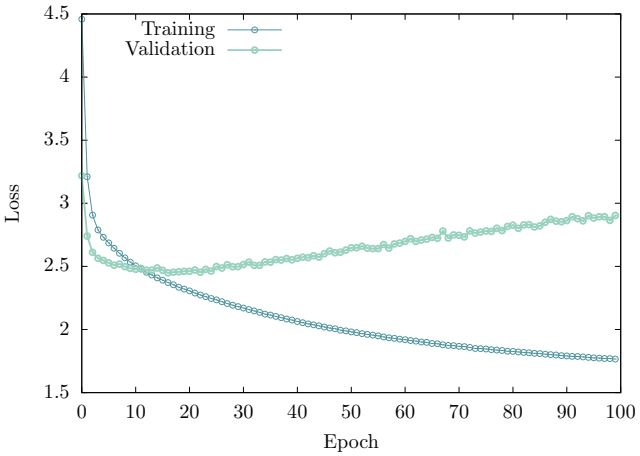

Figure 8: Validation and training loss for the pretrained model with GloVe embeddings.

Pretrained model translation examples

```
The children love listening to stories.
[start] a los niños les encanta escuchar historia [end]

The buildings are small in comparison to the skyscrapers
in New York.
[start] los [UNK] son grandes [UNK] para [UNK] en el
nuevo [end]

He suggested to me that we go to the beach.
[start] Él me habló en esta playa [end]

Tom thinks women in America wear too much perfume.
[start] tom piensa que las mujeres en [UNK] demasiado
[UNK] [end]

I've spent all the money.
[start] he perdido todo el dinero [end]

He thought someone had put poison in his soup.
[start] Él pensó que había [UNK] en la sopa [end]

I knew you'd be in here.
[start] sabía que tú [UNK] aquí [end]

I'm the only one who had to do that.
[start] soy el única que tenía que hacer eso [end]

Tom claimed to be the son of a rich man.
[start] tom [UNK] ser el hijo [end]

Tom is well aware of the problem.
[start] tom es muy [UNK] del problema [end]
```