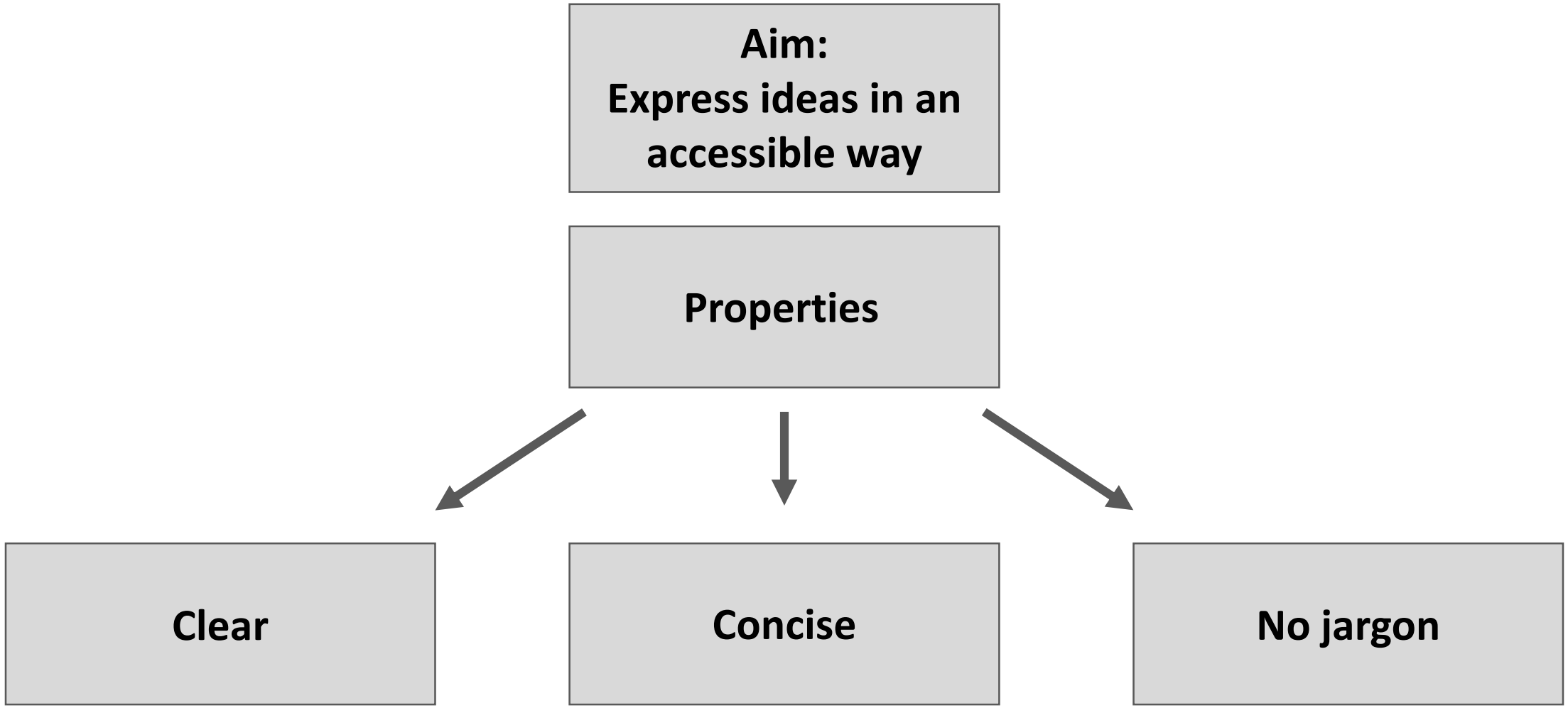
Plain Language Classification

Hannes Körner

04.07.2024

Plain Language



Plain Language

Examples [edit]

Original text	Plain language
High-quality learning environments are a necessary precondition for facilitation and enhancement of the ongoing learning process.	Children need good schools so they can learn well.[17]
Firearm relinquishment is a mandatory condition.	You must hand over your guns.[18]
This temporary injunction remains in effect against both parties until the final decree of divorce or order of legal separation is entered, the complaint is dismissed, the parties reach agreement, or until the court modifies or dissolves this injunction. This injunction shall not preclude either party from applying to the court for further temporary orders, an extended injunction or modification or revocation of this temporary injunction.	You **must** follow this order until the court changes or ends it, your case is finalized or dismissed, or you and your spouse make an agreement. Either spouse may ask the court to change or cancel this order[18] or to issue new orders.
"While we are committed to – and our strategy continues to leverage – our unparalleled global network and footprint, we have identified areas and products where our scale does not provide for meaningful returns. And we will further increase our operating efficiency by reducing excess capacity and expenses, whether they centre on technology, real estate or simplifying our operations."	We are a successful company, but today we announced lay-offs. They will save us money.[19][20]

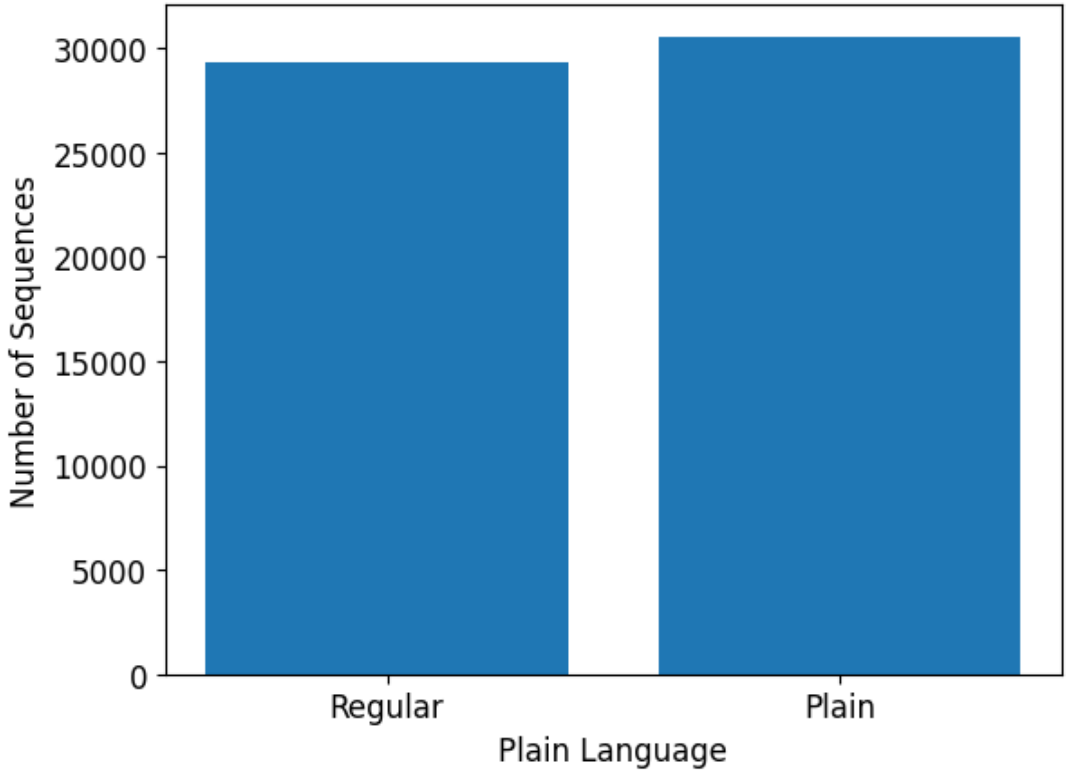
Project Goal

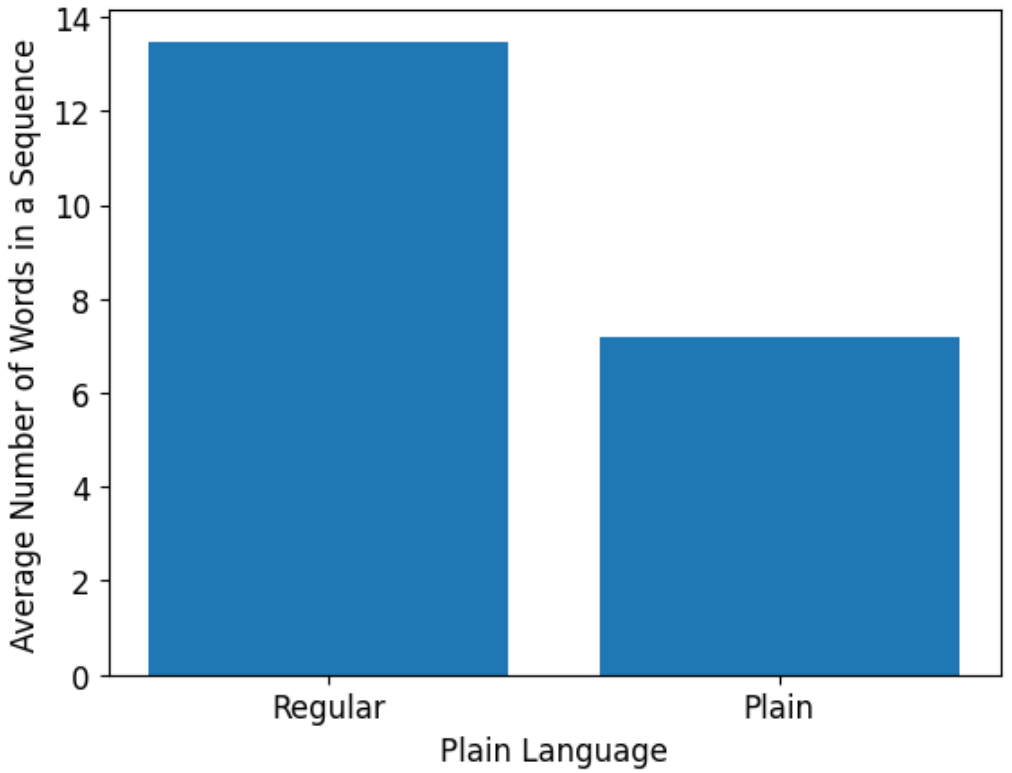
Is a given German text in Plain Language?

Binary text-classification task

Data Corpus from Toborek et al. (2023)

- German Corpus of sentences in regular language and plain language
- Both „Leichte Sprache“ (easy language) and „Einfache Sprache“ (simple language)





Baseline Model – Implementation

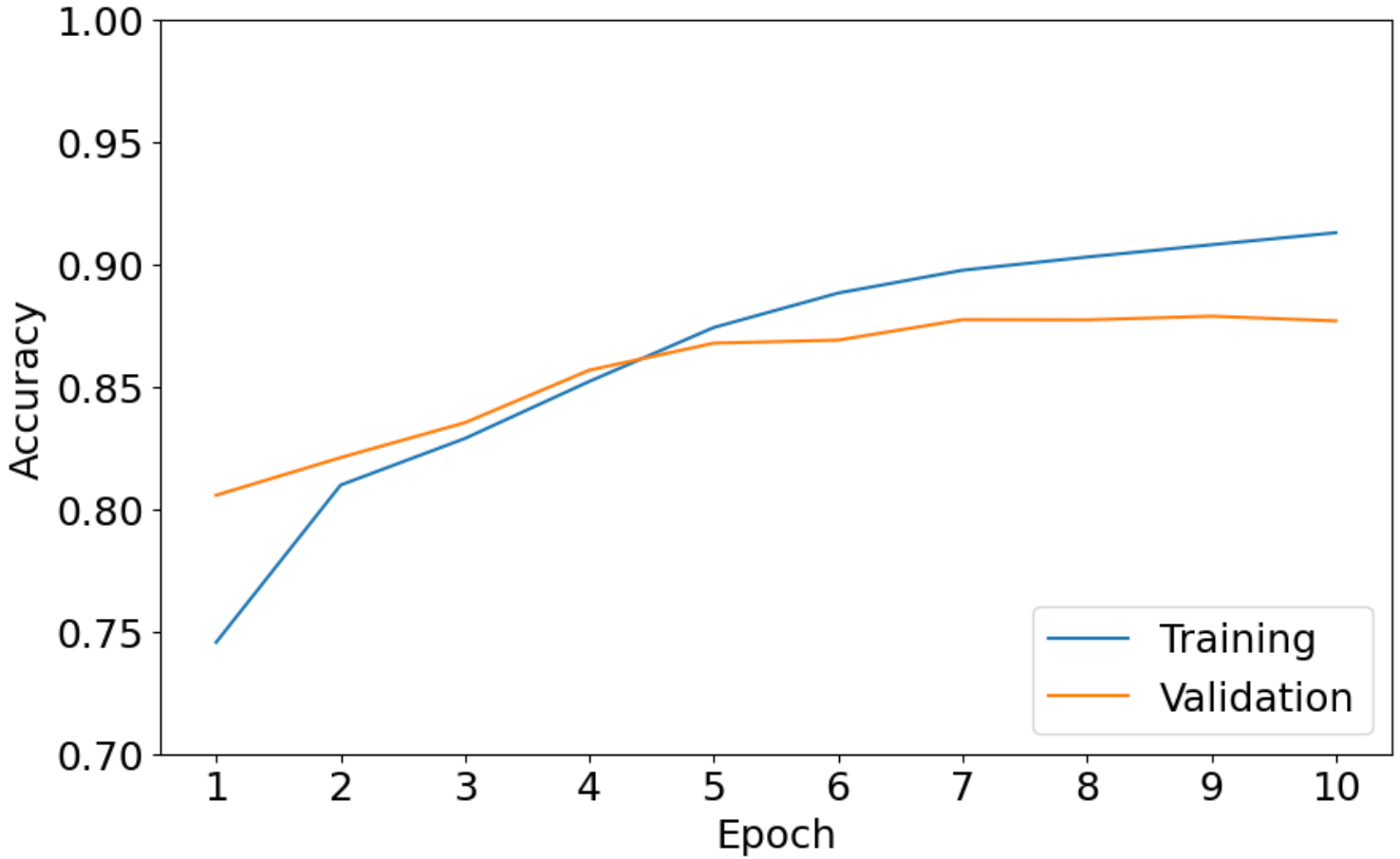
TextVectorization layer (instead of Tokenizer)

Embeddings trained from scratch

1 hidden dense layer

```
vocabulary_size = 10_000
maximum_sequence_length = 12
dimensions_embedding = 32
units_dense = 8

layer_vectorization = TextVectorization(
    max_tokens=vocabulary_size,
    standardize="lower",
    output_sequence_length=maximum_sequence_length,
)

layer_vectorization.adapt(X_train)

model = Sequential([
    Input(shape=(1,), dtype=tf.string),
    layer_vectorization,
    Embedding(
        input_dim=vocabulary_size,
        output_dim=dimensions_embedding
    ),
    GlobalAveragePooling1D(),
    Dense(units=units_dense, activation="relu"),
    Dense(units=1, activation="sigmoid"),
])

model.summary()
```

Baseline Model – Results



Accuracy: 88%

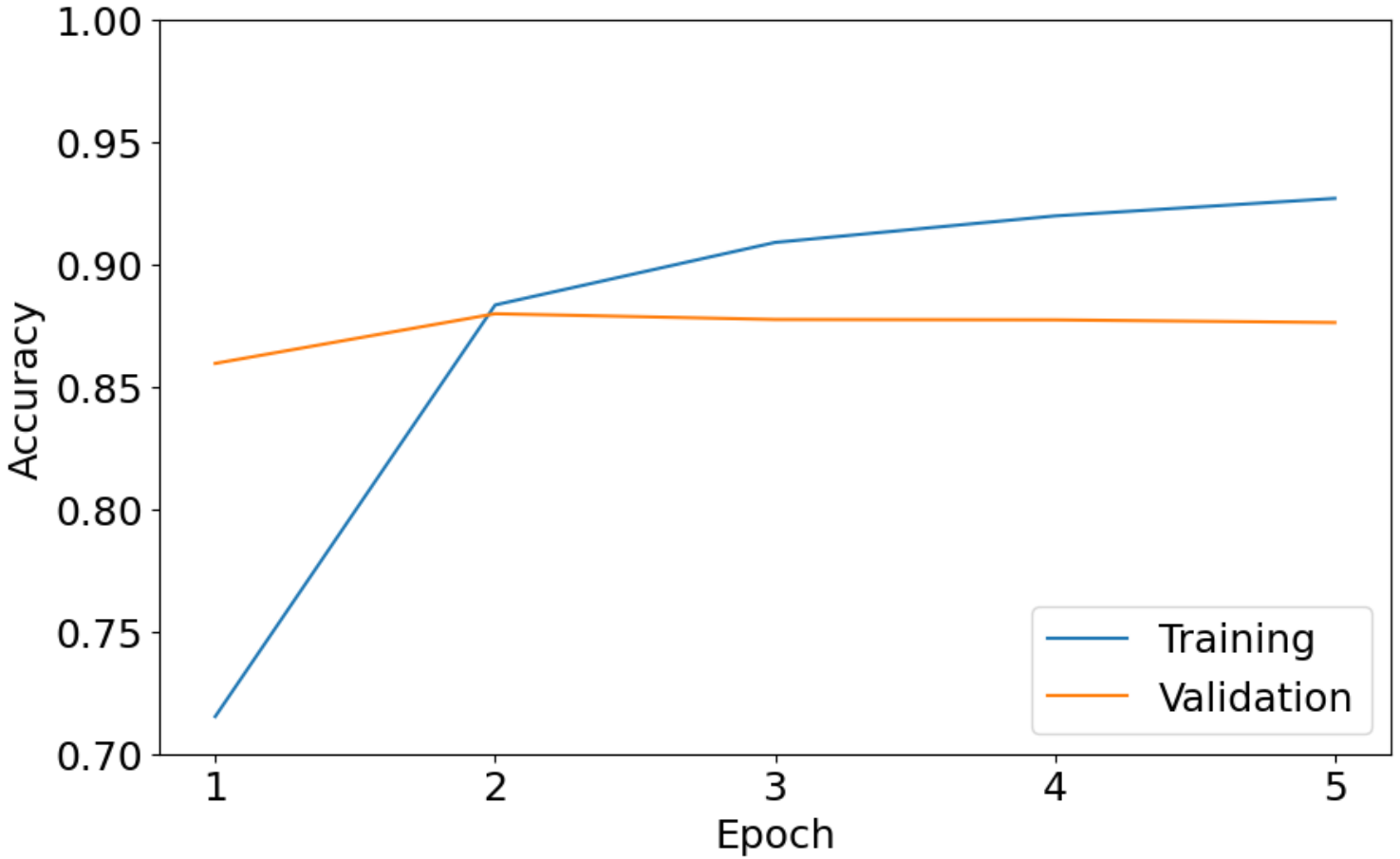
Model 1 (LSTM) – Implementation

3 bidirectional LSTM layers

A dropout layer after the hidden dense layer

```
dimensions_embedding = 32
units_lstm_1 = 64
units_lstm_2 = 32
units_lstm_3 = 8
units_dense = 8

# Define the model architecture
model = Sequential([
    Input(shape=(1,), dtype=tf.string),
    layer_vectorization,
    Embedding(
        input_dim=vocabulary_size,
        output_dim=dimensions_embedding,
        mask_zero=True
    ),

    Bidirectional(LSTM(units=units_lstm_1, return_sequences=True)),
    Bidirectional(LSTM(units=units_lstm_2, return_sequences=True)),
    Bidirectional(LSTM(units=units_lstm_3)),

    Dense(units=units_dense, activation="relu"),
    tf.keras.layers.Dropout(.1),

    Dense(units=1, activation="sigmoid"),
])
```

Model 1 (LSTM) – Results



Accuracy: 88%

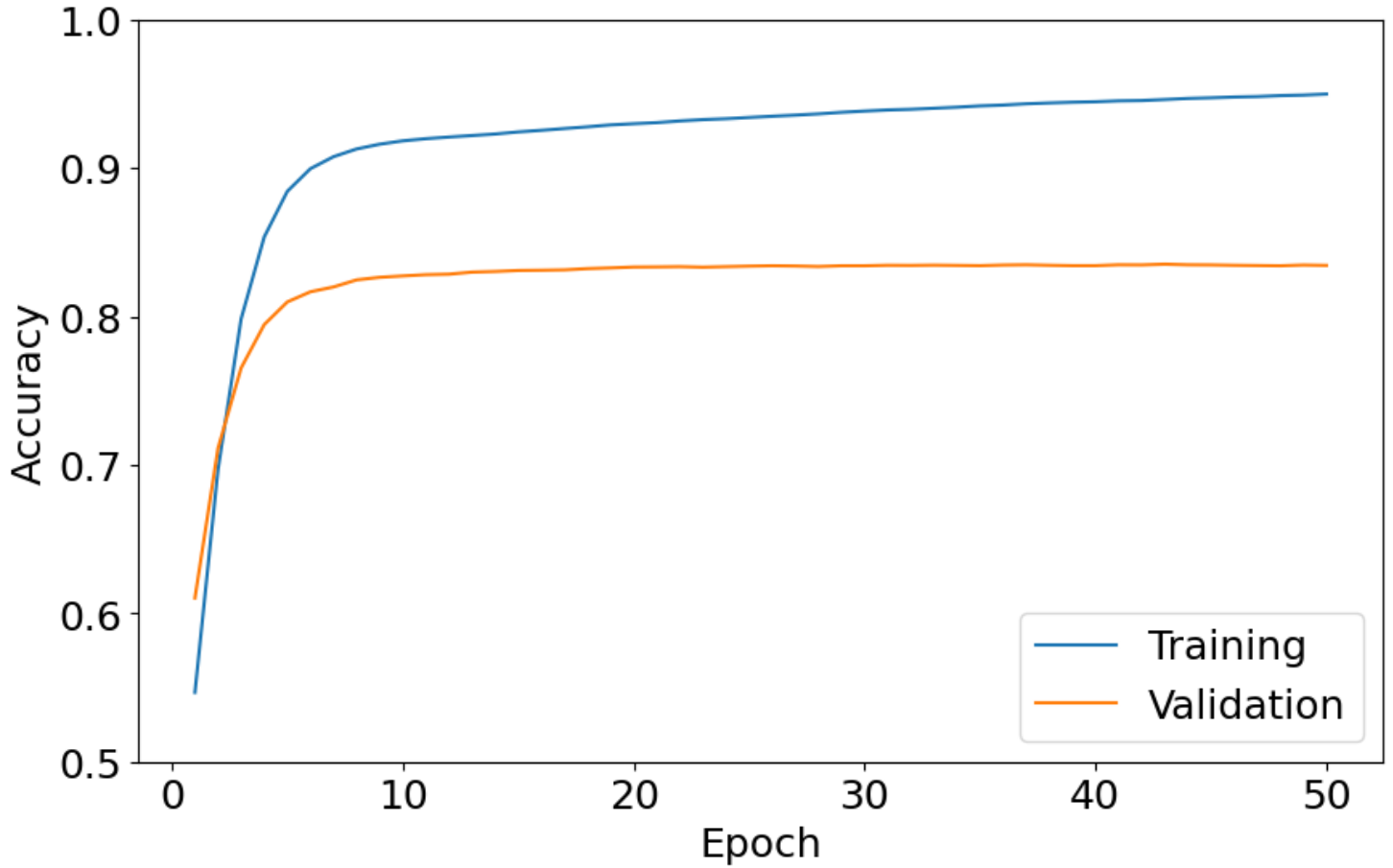
Model 2 (NNLM) – Implementation

Pre-trained embeddings (Neural-Net Language Model)

1 hidden dense layer

```
import tensorflow_hub as hub

embedding_dimensions = 128  # can be 50 or 128

units_dense = 16

hub_layer = hub.KerasLayer(
    f"https://www.kaggle.com/models/google/nnlm/TensorFlow2/de-dim{embedding_dimensions}/1",
    input_shape=[],
    dtype=tf.string,
    trainable=True,
)

model = Sequential([
    hub_layer,
    Dense(units=units_dense, activation="relu"),
    Dense(units=1, activation="sigmoid"),
])
```

Model 2 (NNLM) – Results



Accuracy: 83%

Model 3 (GBERT) – Implementation

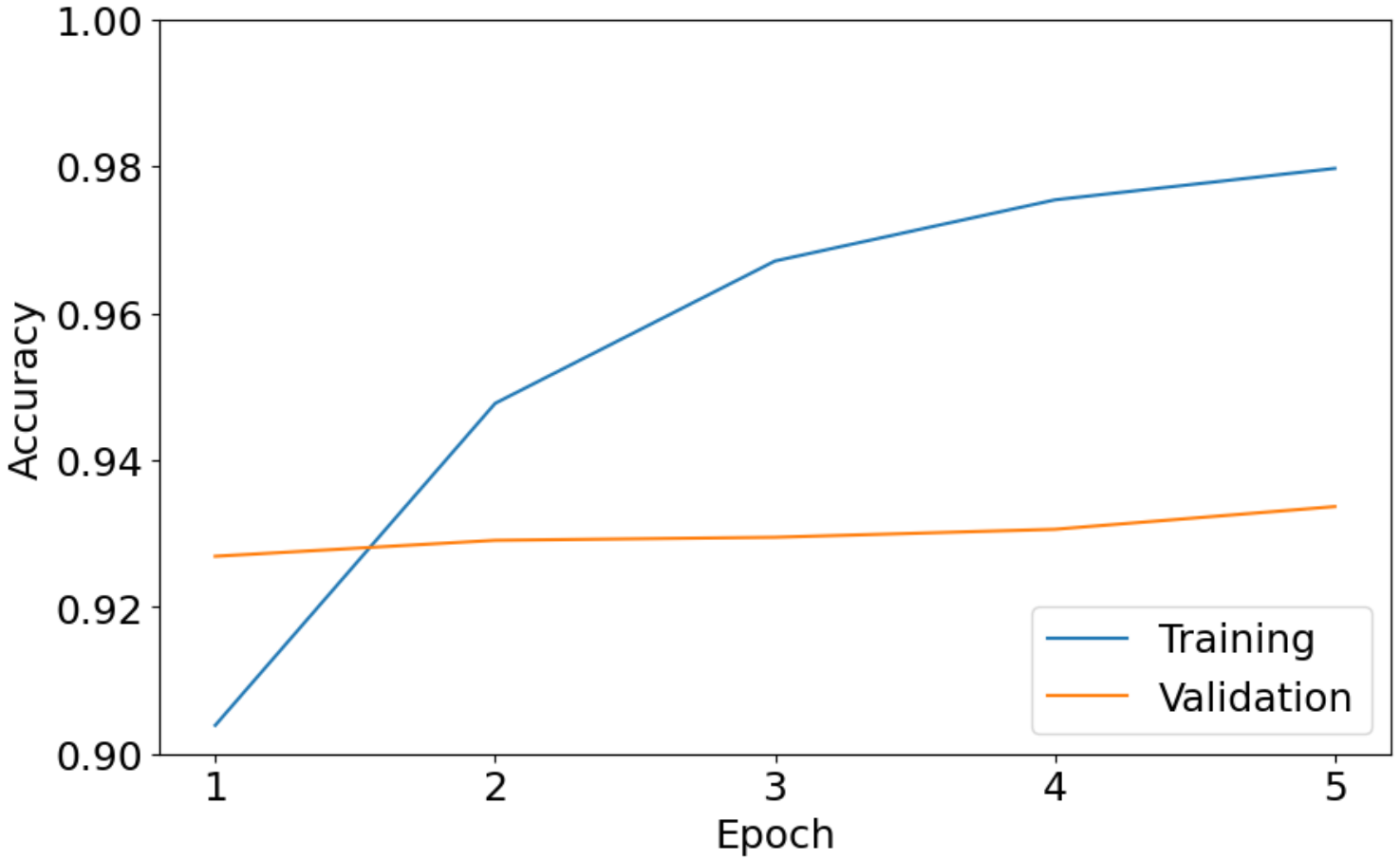
Pre-trained Transformer-based model

No stop-word removal

Longer sequences (30 tokens instead of 12)

```python
import tensorflow as tf
from transformers import AutoTokenizer, TFAutoModelForSequenceClassification

model_name = "deepset/gbert-base"

maximum_sequence_length = 30

tokenizer = AutoTokenizer.from_pretrained(model_name)

def tokenize_text(
    texts: List[str],
    tokenizer: AutoTokenizer,
    maximum_sequence_length: int,
):
    encoded_texts = tokenizer(
        texts,
        truncation=True,
        padding=True,
        max_length=maximum_sequence_length,
    )

    return encoded_texts

encodings_training = tokenize_text(
    texts=texts_training,
    tokenizer=tokenizer,
    maximum_sequence_length=maximum_sequence_length,
)
encodings_validation = tokenize_text(
    texts=texts_validation,
    tokenizer=tokenizer,
    maximum_sequence_length=maximum_sequence_length,
)

model = TFAutoModelForSequenceClassification.from_pretrained(model_name, num_labels=1)
```

Model 3 (GBERT) – Results



Accuracy: 93%

Results – Overview

Model	Accuracy
Baseline	88%
LSTM	88%
Embeddings from an NNLM	83%
Fine-tuned German Bert	93%

Challenges

Getting the data

Exceeding baseline performance

Getting GBERT to run

Outlook

Automated Hyperparameter tuning

Aggregate model predictions across a longer text

Find concrete violations of plain-language rules

Graphical User Interface (e.g., Django/Flask)

Thank you for your attention!