Natural Language Inference

Deep Learning Workshops Project

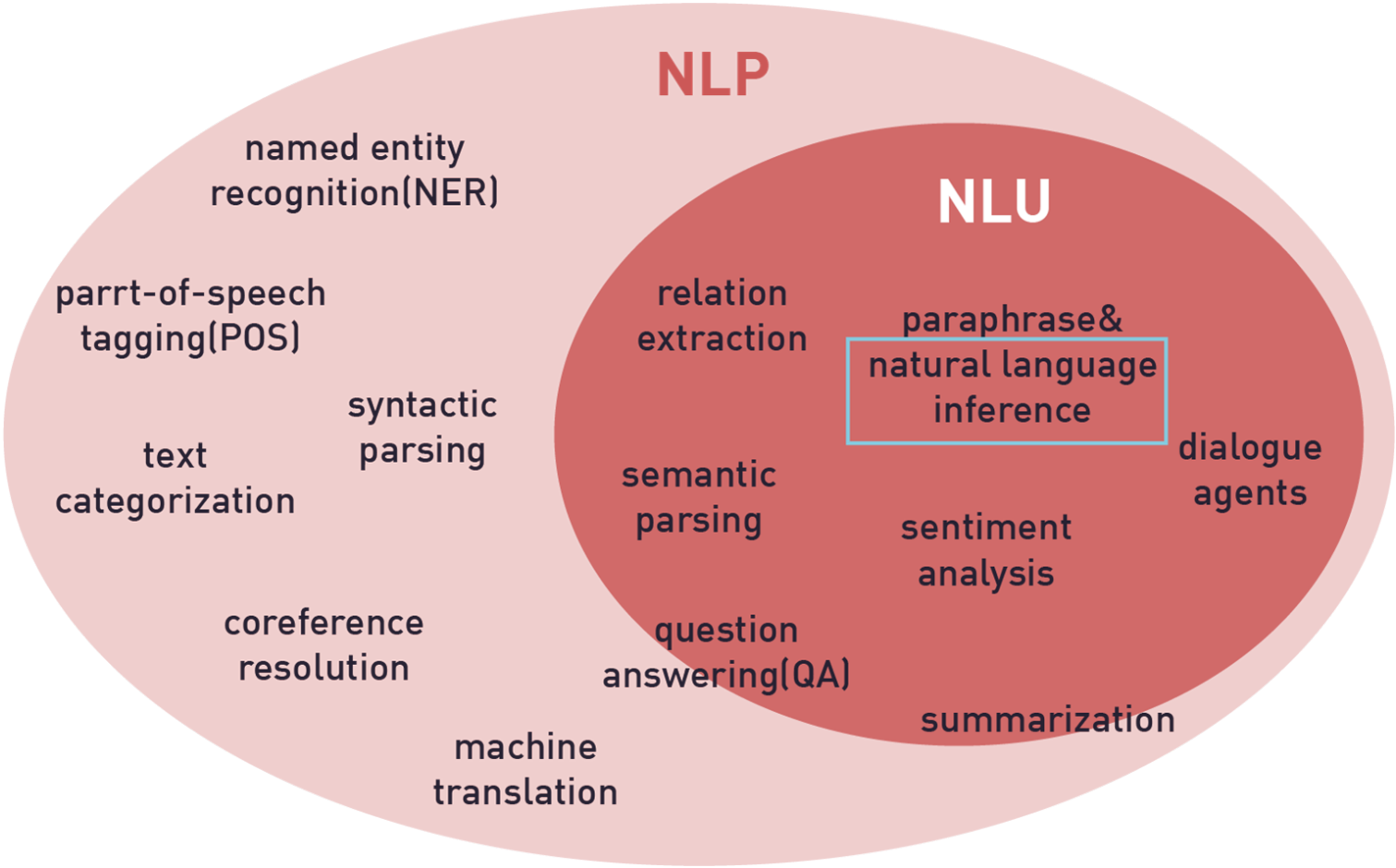
Adrien JAYAT Ryan BELAIB Etienne DE POIX

5th May 2025

Index

Problem Background

Natural Language Inference (NLI) is a kind of **Natural Language Understanding** (NLU) task.



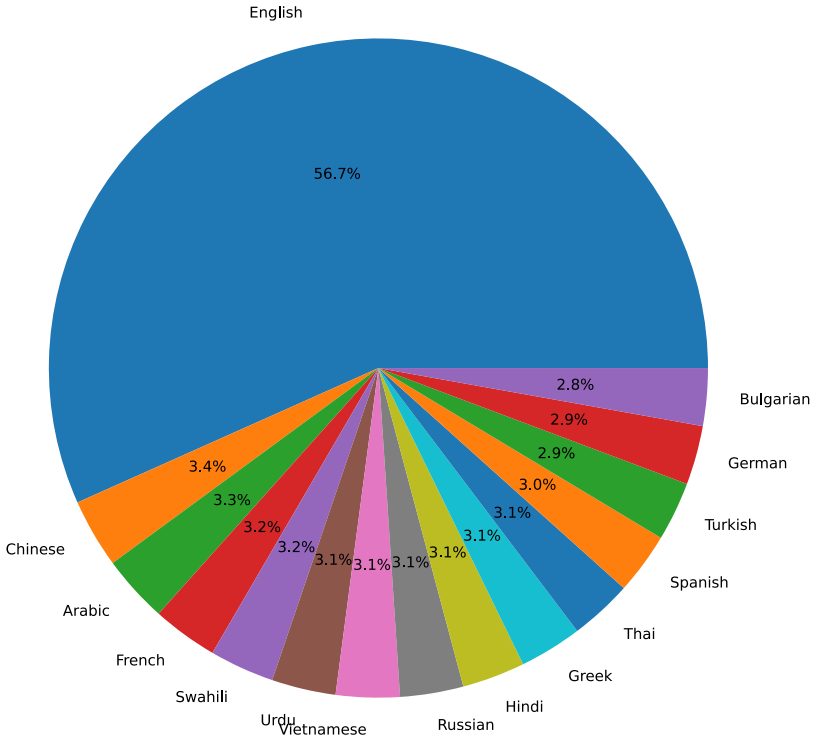
Dataset Instance

Pairs of sentences have a label. It indicates whether the two sentences are in `entailment` (0), `neutral` (1) or in `contradiction` (2).

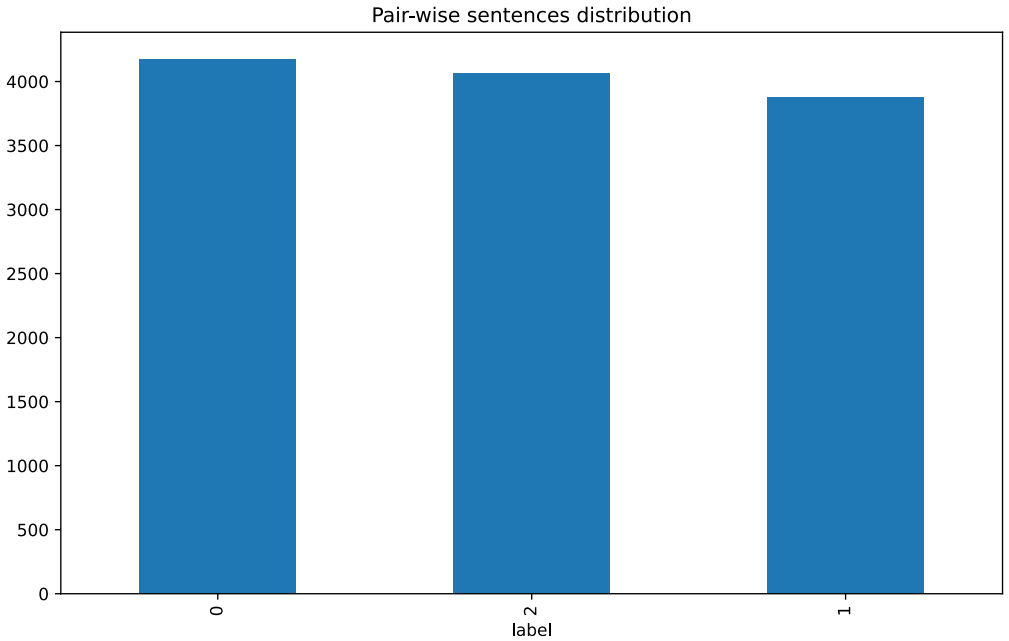
premise	hypothesis	lang_abv	language	label
بکس، کیسی، راہیل، یسعیاہ، کیلی، کیلی، اور کولمبین ہائی اسکول کے دوسرے طلبا کے نام سے بکسوں کو نشان زد کیا جائے گا	کیسی کے لئے کوئی یادگار نہیں ہوگا، کولمین ہائی اسکول کے طالب علموں میں سے ایک جو مر گیا۔	ur	Urdu	2
هذا هو ما تم نصحنا به.	عندما يتم إخبارهم بما يجب عليهم فعله ، فشلت الإدارة في السماح لنا بالدخول إلى الأسرار التجارية.	ar	Arabic	1
et cela est en grande partie dû au fait que les mères prennent de la drogue	Les mères se droguent.	fr	French	0
Она все еще была там.	Мы думали, что она ушла, однако, она осталась.	ru	Russian	1
His family had lost a son and a daughter now.	The son and daughter had lost their father.	en	English	2
Steps are initiated to allow program board membership to reflect the clienteligible comm	There's enough room for 35-40 positions on the board.	en	English	1
C'était probablement la première chose dont je me souvenais de ma petite enfance, et en	C'était l'un de mes premiers souvenirs.	fr	French	0
agencies' operating trust, enterprise and internal service funds) are required to produce	Agencies in financial trouble are usually audited.	en	English	1
Hakuna aliyejua walipokwenda.	Mafiko yao ilikuwa ni siri	sw	Swahili	0
how long has he been in his present position	What length of time has he held the current position?	en	English	0
Il faut habituellement plus de temps pour élaborer le plan d'action.	Ils peuvent élaborer le plan plus rapidement que prévu.	fr	French	2
Research and development is composed of	R&D is made up of.	en	English	0
Then I considered.	I refused to even consider it.	en	English	2
Хакерам или просто увлекающимся, вероятно, не составит труда перевести то, что	Хакеры с удовольствием переводят компьютерный слэнг на нормальный а	ru	Russian	1
Yes, sir.	I will take care of that right away Sir.	en	English	1
It vibrated under his hand.	It hummed quietly in his hand.	en	English	0
Time reports that Harrer denies having known she was.)	Harrer doesn't claim he knows she was.	en	English	0
Managing better requires that agencies have, and rely upon, sound financial and prograı	Agencies that rely on information based on unsound financial information will h	en	English	1
So let me draw a slightly different moral from the saga of beach volleyball as it has evolv	If a village is to be free, Speaker Gingrich believes they should not have a volleyb	en	English	2

Data

The provided dataset contains 12120 premise-hypothesis pairs.



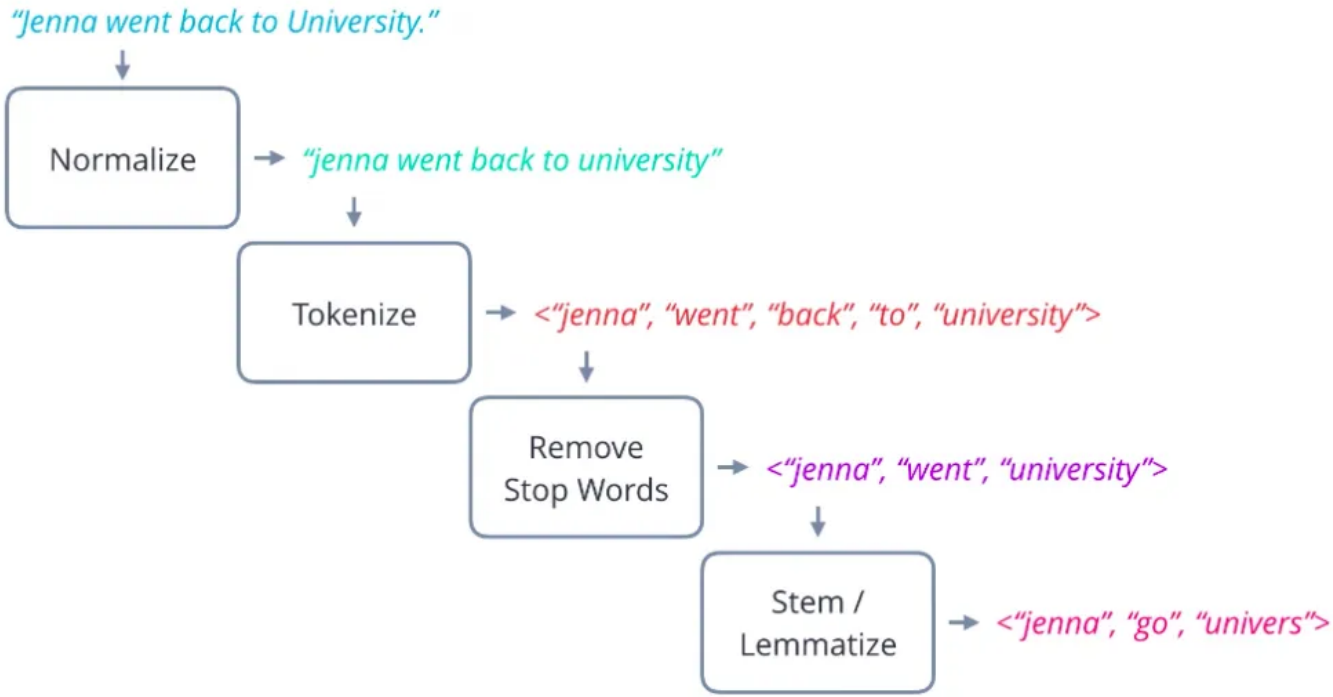
(a) Languages distribution



(b) Labels distribution

Preprocessing

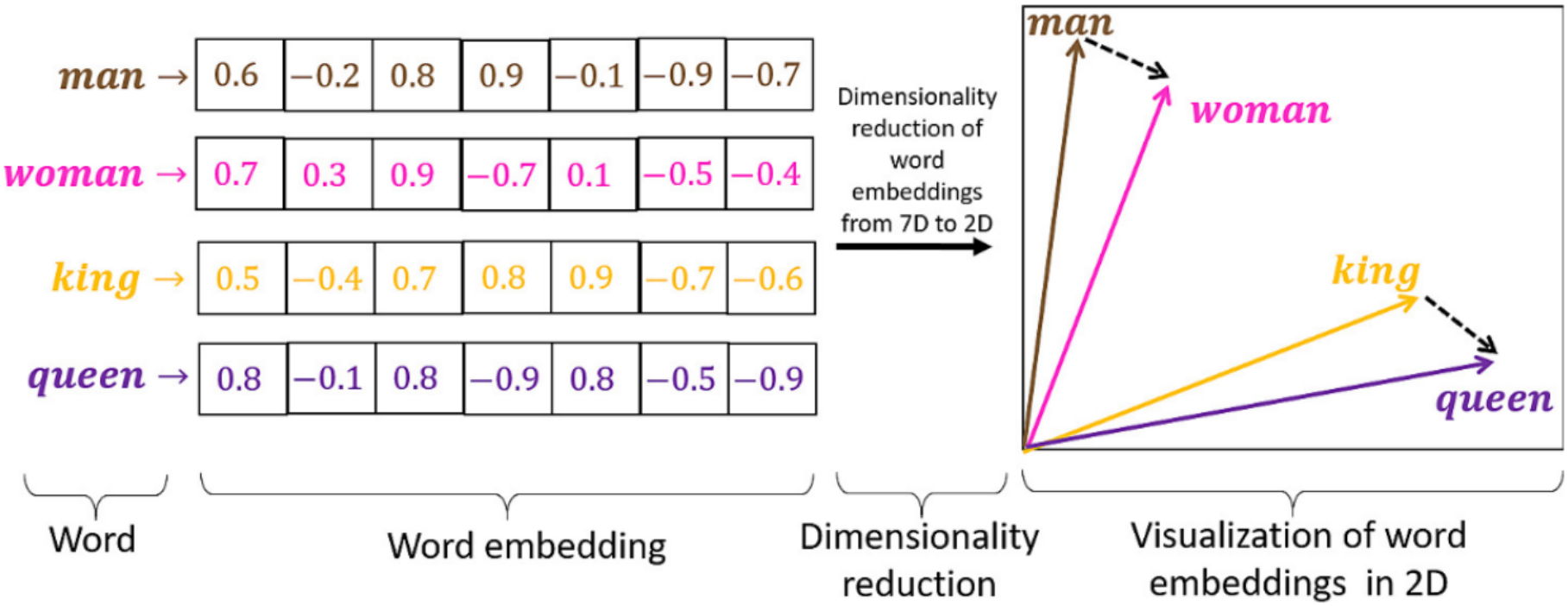
Data preprocessing: tokenization, stop-words removal. Necessary for NLP applications.



Word Embeddings

- An embedding is a vector representation of a word in a low-dimensional space.
- Unlike one-hot encoding, it captures semantic similarities between words.
- **Distributional hypothesis**: words with similar meanings tend to appear in similar contexts.
- **Applications**: sentiment analysis, machine translation, dimensionality reduction.

Embeddings Visualisation



Source: medium.com

Word2Vec

Introduced by Mikolov et al. (2013), Word2Vec learns dense word embeddings from large text corpora.

Two main architectures:

- **CBOW (Continuous Bag of Words)**: Predicts a target word from its surrounding context words.
- **Skip-Gram**: Predicts context words from a given target word.

Objective: Maximize the probability of word-context co-occurrences within a fixed-size window.

Evaluation

We use the **Skip-Gram** architecture with a context window size of 4 to effectively capture the context of each premise and hypothesis.

Word2Vec Limitation: It does not produce sentence-level embeddings (unlike models like BERT with [CLS] tokens).

Solution: Compute sentence embeddings by averaging token embeddings.

Classification: Random Forest with 200 trees.

Limitations of Word2Vec for Entailment

Result: Word2Vec + Random Forest achieves only **32% accuracy**, close to random guessing.

Why this approach fails:

- Word2Vec captures surface-level lexical similarity.
- Sentence embeddings are based on average word vectors, ignoring word order and context.
- Semantically unrelated sentences can appear close in vector space if they share common words.

Key issue: Entailment requires understanding negation, context, and logical structure — which Word2Vec cannot model.

Illustrative Example and Transition

Example Misclassification:

- **Premise:** *The Journal put the point succinctly: "Is any publicity good publicity?"*
- **Hypothesis:** *The Journal asked, "Is this a good political move?"*
- **Predicted:** Contradiction **(Wrong)**
- **Actual:** Neutral

Explanation: Overlapping words like *"The Journal," "asked," "good"* cause embeddings to cluster, misleading the model.

Takeaway: Shallow embedding methods fail to capture deeper semantics.

Next: We turn to **context-aware models** like **BERT**, which handle these complexities more effectively.

Simple Sentence Embedding

Model used: `all-MiniLM-L6-v2`

- Lightweight sentence transformer producing a 384-dimensional embedding per sentence.
- Trained via knowledge distillation from a larger model (BERT-large), maintaining performance while reducing computational cost.

Usage in our pipeline

- Each premise and hypothesis is independently encoded into a dense vector.
- These embeddings are combined (concatenation + absolute difference) to form input for downstream classifiers models

Models and Input Representations

- **Dataset creation**:
  - Sentences encoded using `all-MiniLM-L6-v2`.
  - For each pair: premise $\rightarrow \vec{u}$, hypothesis $\rightarrow \vec{v}$.

- **Two input approaches**:
  - **Approach 1**: Concatenate the two embeddings:

$$x = [\vec{u}, \vec{v}]$$

  - **Approach 2**: Add element-wise absolute difference:

$$x = [\vec{u}, \vec{v}, |\vec{u} - \vec{v}|]$$

- **Models used**:
  - **KDTree**: non-parametric model based on cosine similarity between x and training examples.
  - **Dense Neural Network**: trained on the vector x with softmax output for classification (3 labels).

Training

Training of the dense neural network

Lot of overfitting $\rightarrow$ applying high dropout (0.7)

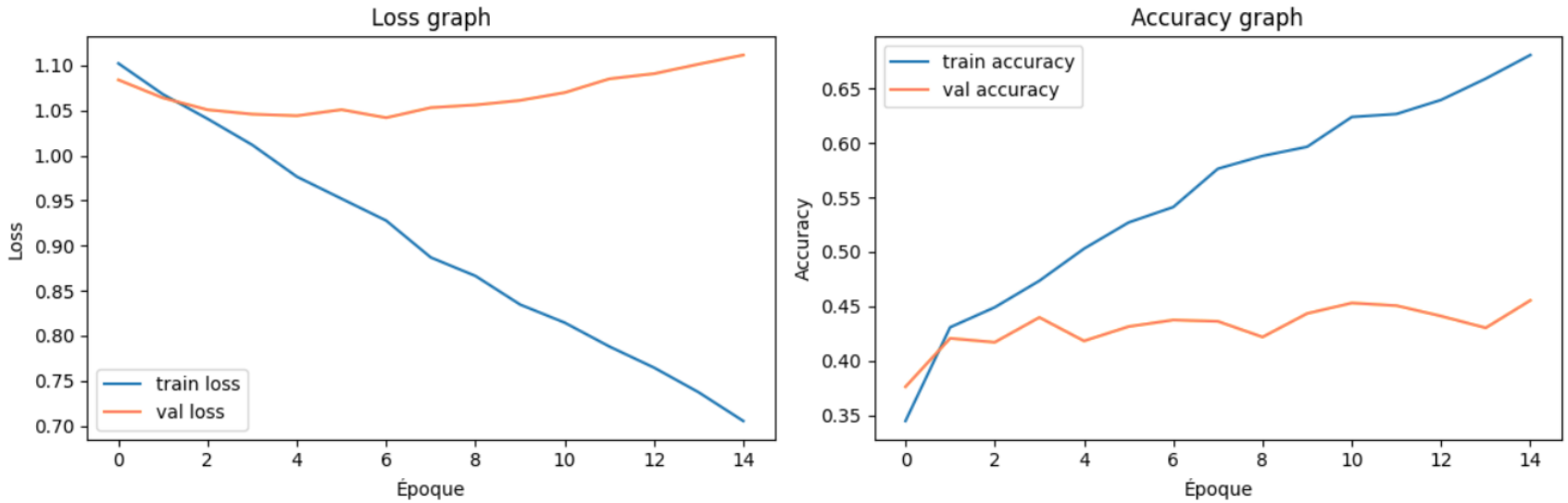


Figure: Training and validation curves

Results

The confusion matrix shows that the model performs well on the "entailment" class, but have much more difficulties to understand the contradictions.

The 2nd approach (with the absolute difference) gave us about 6% more accuracy.

For the results, we got :

- 40% accuracy with the dense model
- 43% accuracy with the KDTree model

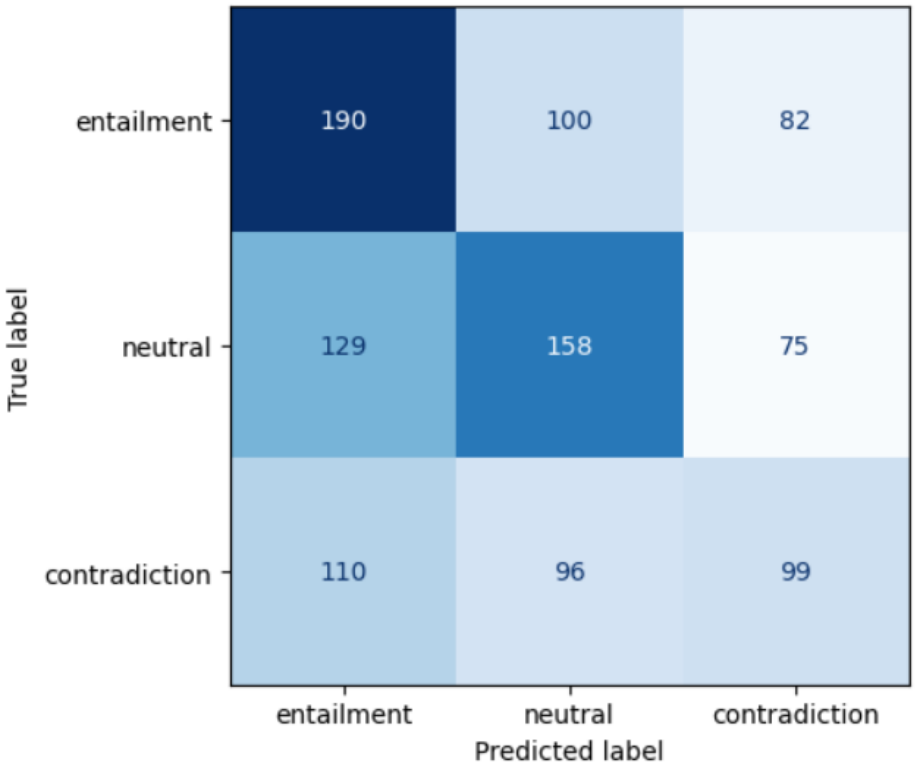
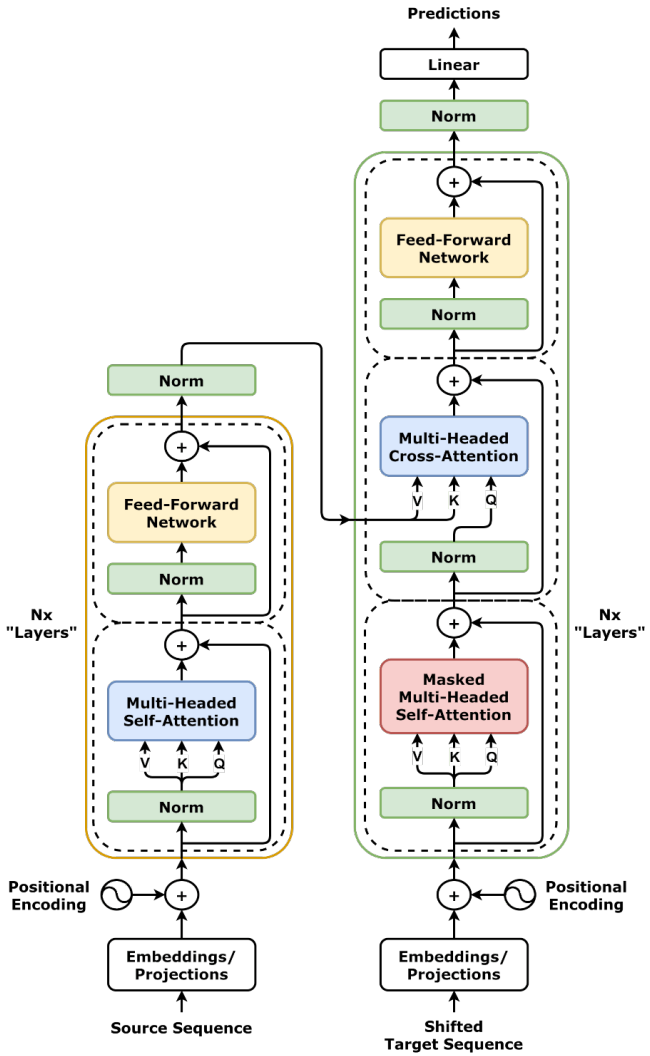


Figure: Confusion matrix of KDTree

Transformers



Transformers have become the state-of-the-art architecture for a wide range of NLP tasks.

Advantages:

- Better parallelization during training, unlike RNNs or LSTMs.
- Better contextual understanding through the attention mechanism.
- Efficient handling of long-range dependencies between words.

Datasets

We introduced two popular datasets for the NLI task published by Facebook AI:

- **MNLI** (Multi-Genre NLI)
  $\rightarrow$ Adds a genre label for others NLU tasks.
- **XNLI** (Cross-lingual NLI), an evaluation corpus in 15 languages

Dataset	Train	Validation	Test
Challenge (provided)	12k[1]	✗	5k
MNLI (English)	393k	10k	10k
XNLI (Multilingual)	5.9M[2]	38k	75k[3]

Table: Number of sentence-pairs for each datasets split

[1] The provided train set was not used for this part.
[2] Machine-translated MNLI training set into 14 languages.
[3] 7.5k pairs collected in English and professionally-translated into 14 languages.

Fine-tuning

Fine-tuning is the process of adapting a pre-trained model to a new, task-specific dataset.

It leverages advantages of **Transfer Learning:**

1. Better performance
2. Reduced training time: Shortens training by starting with an already-learned model.
3. Lower data requirements: Needs less data, as the model has learned general features beforehand.

Unleash the Power!

→ We set up a remote Jupyter server (via SSH) connected to 4 GPUs `NVIDIA SXM4 A100 80GB`, providing up to 2500 TFLOPS.

XLM-RoBERTa

For multilingual text classification, these two models are widely used:

- Facebook `XLM-RoBERTa` (encoder-only)
- Google `mT5` (encoder-decoder)

Model	Parameters	Layers
XLM-RoBERTa Base	270M	12
XLM-RoBERTa Large*	550M	24
XLM-RoBERTa XL	1.3B	32
XLM-RoBERTa XXL	2.5B	48
mT5 Base	300M	12
mT5 Large	770M	24
mT5 XL	3.7B	32
mT5 XXL	13B	48

Table: Comparison of XLM-RoBERTa and mT5 model sizes

Classification Head

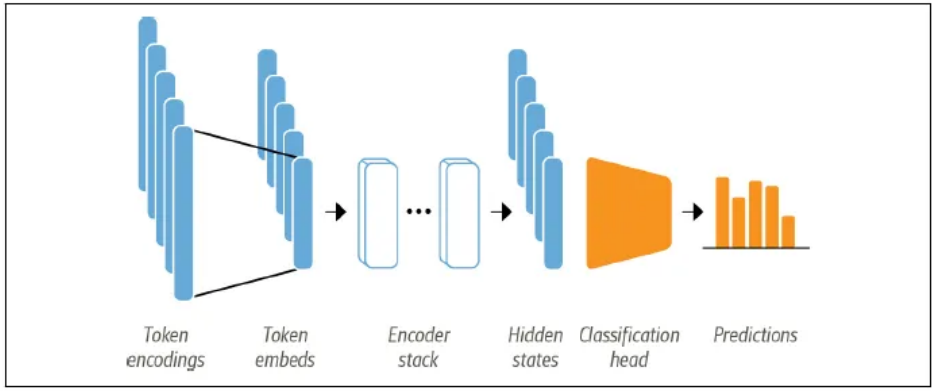


Figure: Architecture used for classification with an encoder-based transformer

Source: medium.com

We tried to build a custom classification head with:

- Average pooling over token embeddings instead of relying on the `[CLS]` token (first `<s>` token in RoBERTa)
- GELU activation and Dropout

→ Slightly worse results than the default classification head.

Training

The data has been pre-processed using the `XLM-RoBERTa` tokenizer.

Training Hyperparameters:

- Batch size: 64
- Epoch: 1 (See [3])
- Learning Rate: $2e-5$
- Mixed precision: `bfloat16` (supported on A100 GPUs)
- Dataset: XNLI (15 languages)
- Evaluation metric: accuracy

$\rightarrow$ Trained using the `Trainer` API from the `transformers` library.

Very very long...

It took **8 hours** to train on 4 GPUs `NVIDIA SXM4 A100 80GB` !
... and probably several weeks on standard GPUs.

Why BFloat16 as a Floating Point format?

Component	FP16 Format	BF16 Format
Sign bit	1 bit	1 bit
Exponent bits	5 bits	8 bits
Fraction bits	10 bits	7 bits

- **16-bit format:** reduce memory usage and computation costs.
- **Higher range**: it uses the same 8-bit exponent as IEEE 32-bit float (single precision). This helps reduce the risk of underflows and overflows.
- The precision loss due to the fewer fractions bits is often negligible for the model weights.

Training History

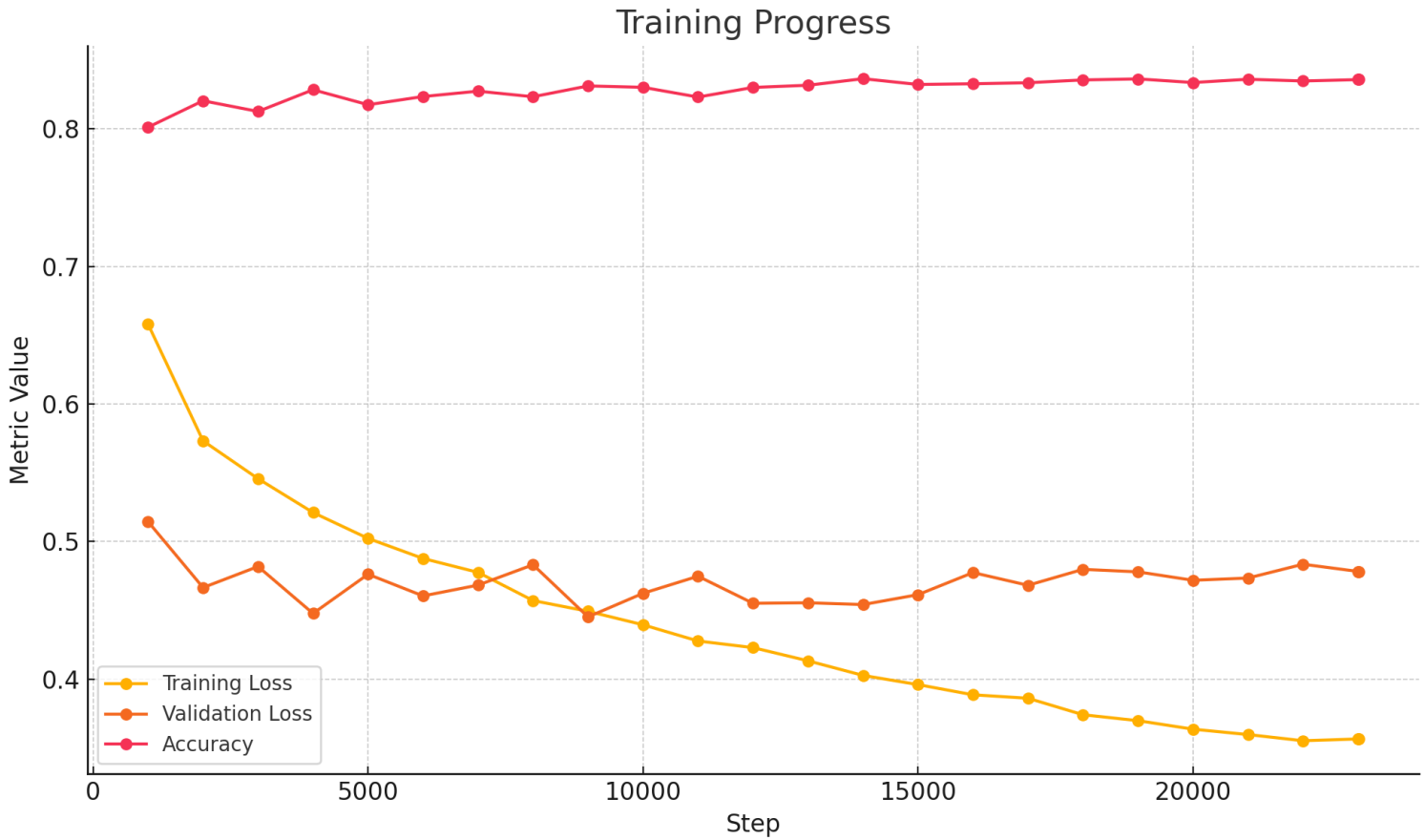


Figure: Training metrics (loss, accuracy) over steps

Kaggle Challenge

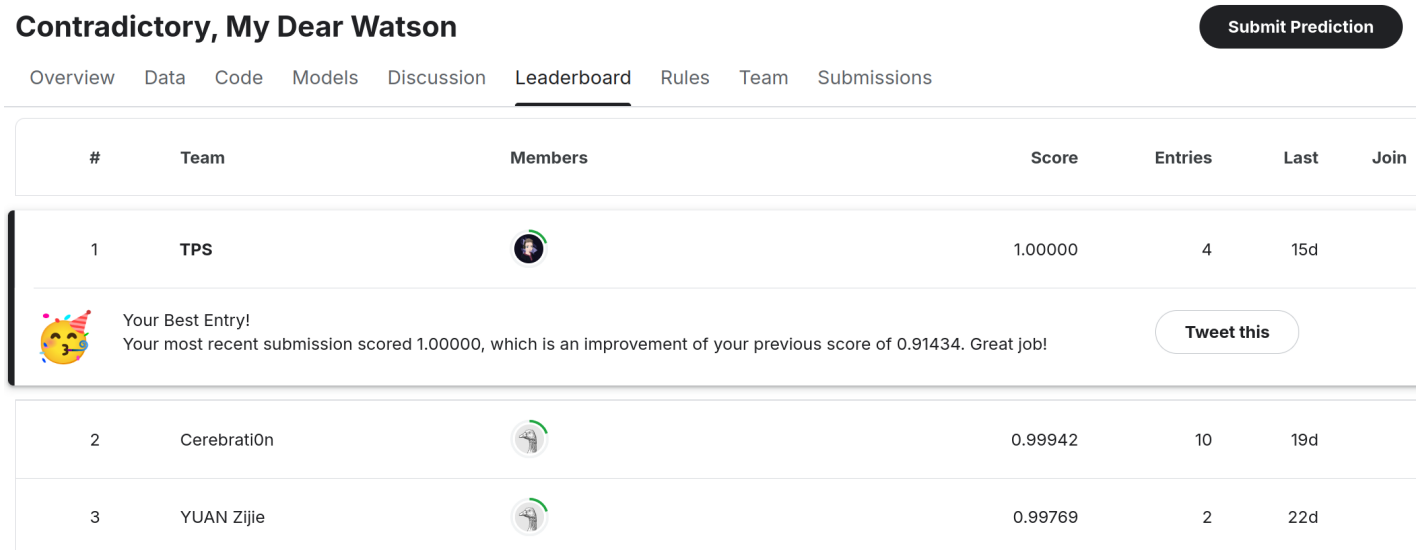


Figure: Kaggle Challenge Leaderboard

We killed it!

We achieved a perfect score... So each team member will receive **$1000**!

Unfortunately, no...[4]

[4] The challenge test set is biased, consisting of a mix of MNLI and XNLI test and validation splits, and we simply overfitted a model to achieve this performance.

Results

- Evaluating our finetuned model on the **challenge** dataset
- Kaggle test set accuracy: **0.86**

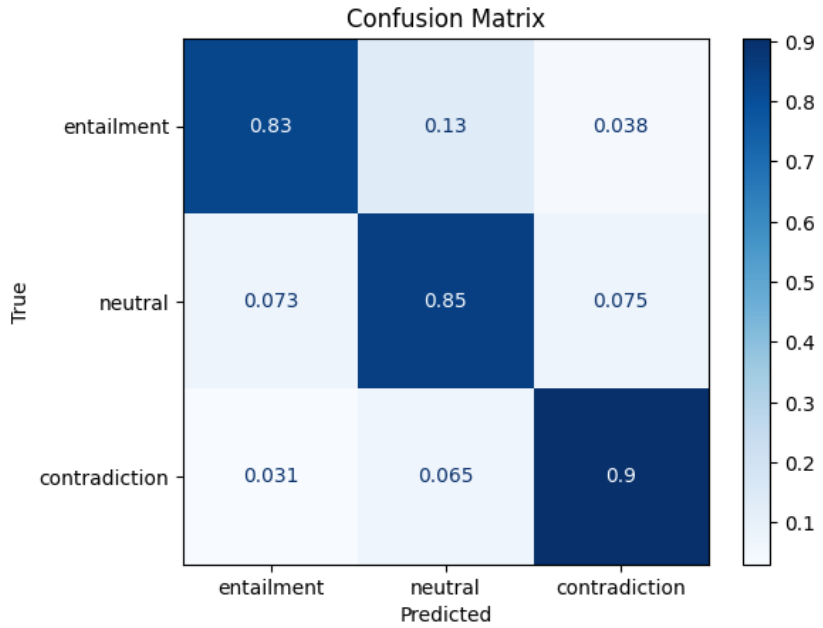


Figure: Confusion matrix

XNLI test set

de	el	en	es	fr	hi	ru	sw	ur	...	avg
0.85	0.85	0.89	0.85	0.84	0.81	0.83	0.77	0.77	...	0.84

Global accuracy: **0.84** (State-of-the-art mT5 model: 0.87)

Attention Visualization

- Example of inference on a premise-hypothesis pair (entailment).
- Use `bertviz` library to inspect attention heads in each layer.
- See how "Some" token attends to "multiple", "male", and "s" tokens.

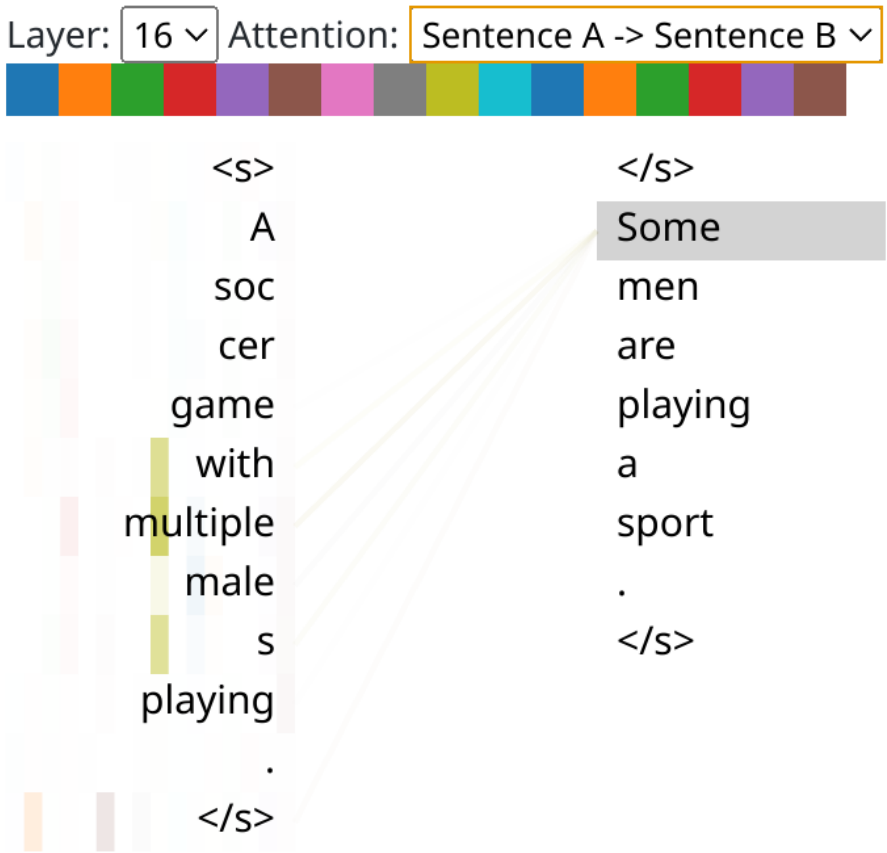


Figure: Attention heads in the 17th layer

Inference Optimization

- Converted model to ONNX format, which is an open source, cross-platform format.
- Applied dynamic quantization using ONNX Runtime to reduce model size and improve speed.
  $\rightarrow$ Model weights and biases are quantized to 8 bits.
- **200x speedup** on CPU inference:

Backend	Inference Time on 1 example (ms)
PyTorch	910 ms $\pm$ 662 ms
ONNX Runtime	4.46 ms $\pm$ 14.4 µs

Zero-Shot Classification

Goal: Test the capabilities of our fine-tuned model on an unseen language.

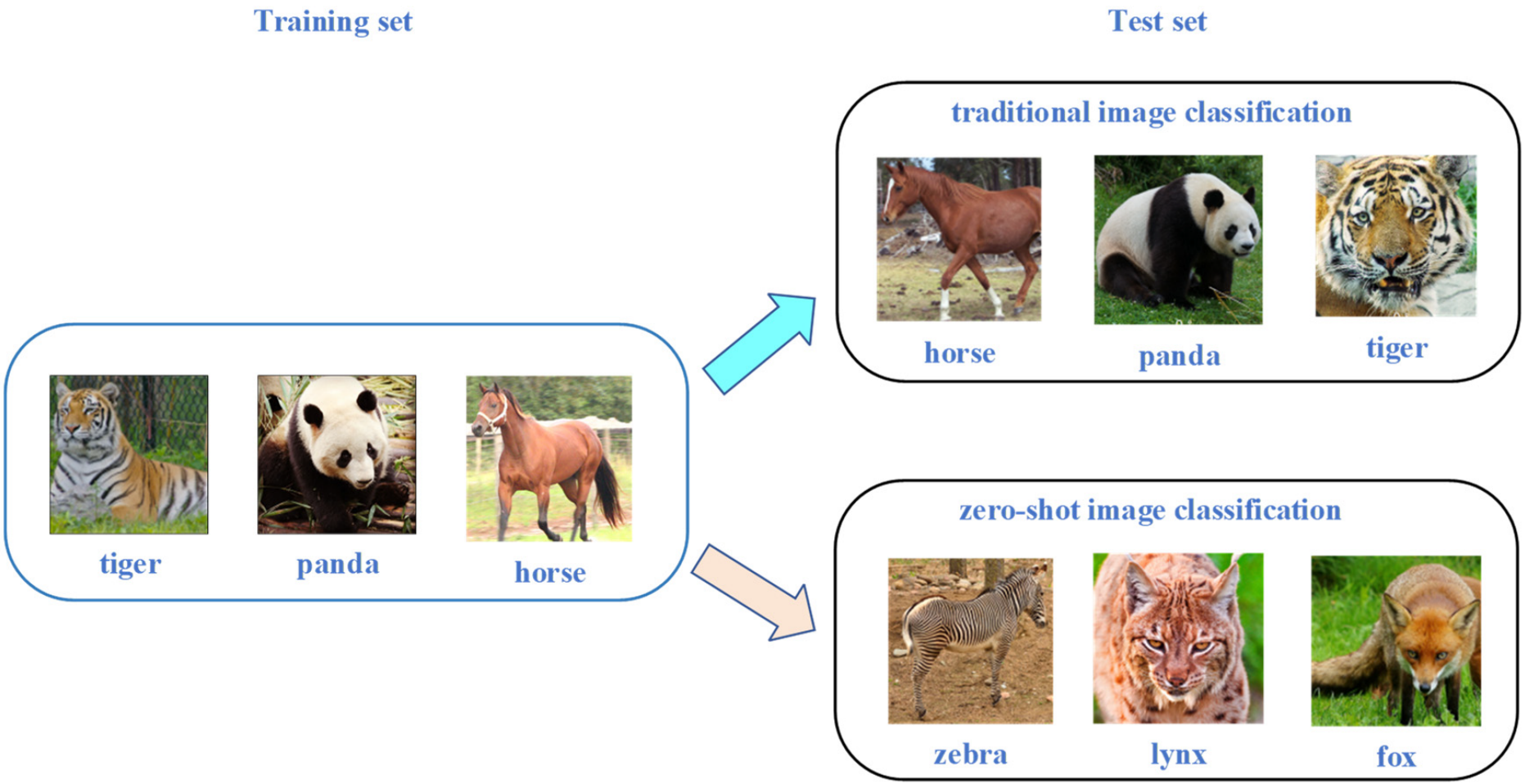
Zero-Shot Classification:

- The model can classify text without needing to be trained on specific labeled data for the target task.
- It leverages prior knowledge learned from other tasks or languages, enabling it to generalize to unseen data.
- In this case, we apply it to a new language that the model has not encountered during training.

Why it's useful:

- Reduces the need for large labeled datasets in the target language.
- Allows the model to perform tasks in multilingual contexts or with new data.

A Visual Example



Source: mdpi.com

Motivation for Zero-Shot Classification

We chose **Korean** for testing our model's zero-shot capabilities due to the following reasons:

- **Unseen Syntax and Typography:** Korean syntax and typography were not present in the data used to fine-tune our model.
- **Challenge:** The model has not been exposed to Korean language patterns during finetuning.

KorNLI Dataset:

- A Korean Natural Language Inference (NLI) dataset.
- Contains 942,854 training examples (auto-translated) and 7,500 evaluation examples (manually translated).

Purpose: Evaluate how well our fine-tuned model can generalize to Korean, despite its lack of exposure during training.

Evaluation Results on Korean (KorNLI)

Performance on Korean (KorNLI) dataset:

Class	Precision	Recall	F1-Score
Entailment	0.80	0.95	0.87
Neutral	0.91	0.73	0.81
Contradiction	0.86	0.88	0.87
Macro avg	0.86	0.85	0.85
Weighted avg	0.86	0.85	0.85

Accuracy: 85%

Conclusion:

- The model performs surprisingly well on an unseen language.
- Likely due to the pre-training of **XLM-Roberta** on Korean data.
- Fine-tuning on the NLI task further enhanced its performance.

Example of Misclassified Instance

Misclassified Sentence Pair:

- **Premise:** 1997년 현대 유니콘스는 선수층 빈약함이 여실히 드러나게 되어 정규 시즌 6위를 기록한다
  (In 1997, the Hyundai Unicorns' weakness in player base was clearly revealed, and they finished in 6th place in the regular season)
- **Hypothesis:** 현대 유니콘스는 1997년 정규 시즌 순위에 들지 못한다
  (The Hyundai Unicorns failed to qualify for the 1997 regular season)

Real Label: 2 (Contradiction)
Predicted Label: 0 (Entailment)

Next Step: Investigating token influence using the **Integrated Gradients** method to understand which tokens contributed to this misclassification.

Interpreting Predictions: Integrated Gradients

Integrated Gradients (IG) is a method used to interpret the predictions of deep learning models, particularly for classification tasks.

Why Use Integrated Gradients?

- Deep models like BERT are often considered black-box models due to their complexity.
- IG provides a way to understand which features (e.g., words in a text) contribute most to a given prediction.
- Helps answer questions like: "Why did the model classify this as positive sentiment?" or "Which words influenced this classification?"

We'll use this method to understand the misclassification of our example instance.

Steps in Integrated Gradients

Step 1: Interpolate the Baseline Input

- The baseline input is typically an empty or neutral version of the input (e.g., all-zero embedding for text).
- Gradually interpolate the baseline input towards the original input.

Step 2: Calculate the Gradients

- For each interpolated input, compute the gradient of the model's prediction with respect to each input feature.

Step 3: Accumulate the Gradients

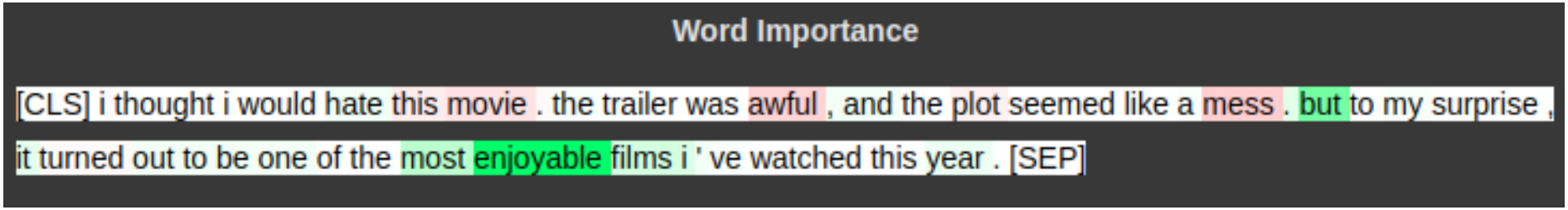
- Accumulate the gradients across all steps using Riemann sums:

$$\mathrm{IG}(x) = \sum_{i=1}^{m} \frac{\partial f(x')}{\partial x_i} \cdot \frac{(x - x')}{m}$$

This gives us the attribution score for each feature.

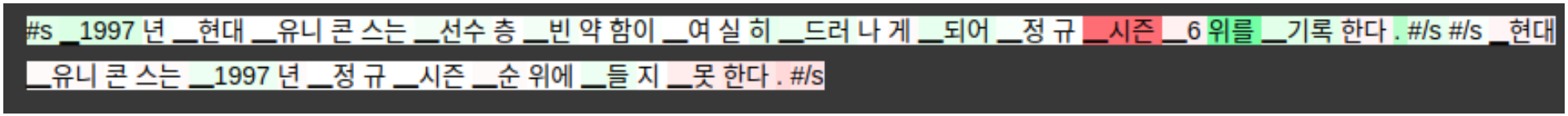
An Example

The tokens that have a positive contribution to the model's prediction are highlighted in green. Meanwhile, the tokens that have negative contributions to the model's prediction are highlighted in red.



Results

Attribution Results: The word "시즌" (season) contributed negatively to the model's prediction, while "위를" (6th place) contributed positively.



Possible Explanations for Word Attribution

- "시즌" (season):
  - Neutral word, typically describing time or schedule, not directly linked to contradiction.
  - Less impactful in identifying contradictions between premise and hypothesis.
- "위를" (6th place):
  - Specific ranking, indicative of outcomes or results, highly relevant for contradiction classification.

Contextual Impact: "시즌" (season) is versatile and can appear in both neutral and contradictory contexts, reducing its discriminative value. "위를" (6th place) is definitive thus strengthening its role in identifying relationships like contradiction or entailment.

Conclusion

Approach	Model	Accuracy
Word-level	Word2Vec + Random Forest	32%
Sentence embeddings	all-MiniLM + KDTree	43%
	all-MiniLM + Dense NN	40%
Transformers	XLM-RoBERTa Large	**86% (Challenge)**
	fine-tuned on XNLI	84% (XNLI)
Zero-Shot	XLM-RoBERTa on KorNLI	85%

Key takeaways

- Context-aware models outperform shallow embeddings.
- Zero-shot NLI is feasible for unseen languages.
- Integrated Gradients aid in model interpretation.

Further improvements

- Curriculum Learning: gradually improves the difficulty of examples during training. [4]

References

- Kaggle. *Contradictory, My Dear Watson*. https://www.kaggle.com/competitions/contradictory-my-dear-watson/.
- A. Conneau et al., *XNLI: Evaluating Cross-lingual Sentence Representations*. https://aclanthology.org/D18-1269/.
- A. Komatsuzaki. *One Epoch Is All You Need*. CoRR, 2019. http://arxiv.org/abs/1906.06669.
- F. Christopoulou, G. Lampouras, I. Iacobacci. (2022). *Training dynamics for curriculum learning: A study on monolingual and cross-lingual NLU*. ACL, 2022. https://doi.org/10.18653/v1/2022.emnlp-main.167

Thank you
for your (human) attention