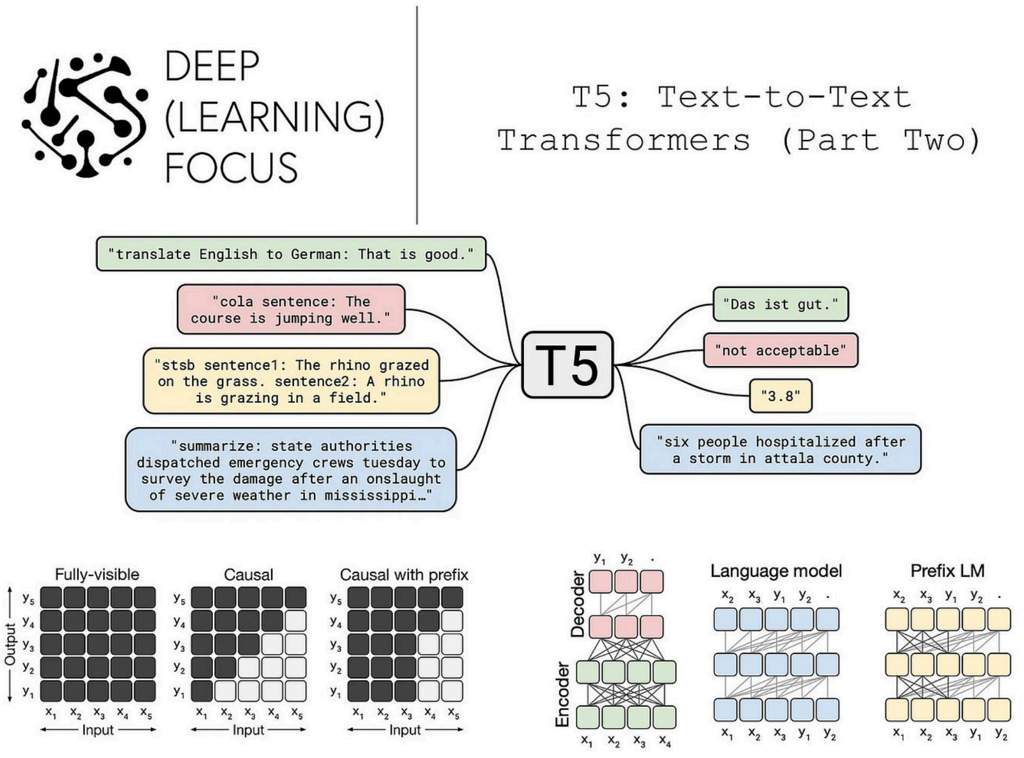
# NLP Project Report

*Fine-Tuning T5 model for Coreference task*


DEEP (LEARNING) FOCUS
T5: Text-to-Text Transformers (Part Two)
"translate English to German: That is good."
"cola sentence: The course is jumping well."
"stsb sentence1: The rhino grazed on the grass. sentence2: A rhino is grazing in a field."
"summarize: state authorities dispatched emergency crews tuesday to survey the damage after an onslaught of severe weather in mississippi…"
T5
"Das ist gut."
"not acceptable"
"3.8"
"six people hospitalized after a storm in attala county."
Fully-visible
Causal
Causal with prefix
Output
Input
Decoder
Encoder
Language model
Prefix LM

**Chinmay Ramdas Rao**

2022A7PS0106G

**Garvit Chittora**

2021B4A61876G

**Trisha Hitesh**

2021B4A72157G

## INTRODUCTION

Natural Language Processing (NLP) is a field of artificial intelligence that enables machines to understand, interpret, and generate human language. It encompasses a variety of tasks, including sentiment analysis, text summarization, machine translation, and question answering.

Coreference resolution is an important NLP task. It involves identifying when different expressions in a text (e.g., pronouns, names, or phrases) refer to the same entity. In this particular task, we are determining a particular entity a pronoun refers to. For our task, we use the T5 model. T5 model is a powerful NLP model that reframes all tasks into a text-to-text format. It is pretrained on the large-scale "Colossal Clean Crawled Corpus" (C4), and is used to acheive state-of-the-art results across several benchmarks by fine-tuning on specific datasets.

## OBJECTIVE

The goal of this assignment is to fine-tune the T5 model, a versatile text-to-text transformer, on the Winograd Schema Challenge (WSC) dataset. The WSC dataset consists of challenging coreference resolution tasks.

To achieve this, the WSC dataset will first be preprocessed to align with the T5 model's text-to-text format, where inputs are crafted as sequences containing the ambiguous pronoun, the candidate entity and its surrounding context. The output is a label, either True or False. For example :

- **Input**: *"In the sentence 'The trophy doesn't fit in the suitcase because it is too small,' Does 'it' refer to 'trophy'?"*
- **Output**: *False*

The T5 model is then fine-tuned with optimal hyper-parameters. Specific attention will be given to optimizing parameters such as learning rate, batch size, number of epochs, warmup steps, and sequence length. These hyperparameters will be tuned to balance training efficiency with generalization on unseen data.

Performance will be evaluated by measuring accuracy on the validation set, reflecting the model's ability to correctly resolve coreferences. This report will detail each step, from dataset preprocessing to model evaluation, highlighting the challenges encountered and solutions implemented during the process.

## DATASET

The dataset used for this project is the **Winograd Schema Challenge (WSC)** dataset, a part of the **SuperGLUE** benchmark suite. SuperGLUE is a collection of challenging natural language understanding tasks designed to evaluate AI models' ability to reason and comprehend complex language structures. The WSC dataset, in particular, focuses on **coreference resolution**, where the goal is to resolve ambiguous pronouns in sentences by determining the correct antecedent. Each instance in the dataset contains a sentence, the pronoun which needs to be resolved, a candidate entity(a noun) and the label whose value depends on whether the pronoun is resolved to the candidate noun.

To align the WSC dataset with the T5 model's text-to-text transfer framework, a preprocessing function was developed to reformat each data point into input-output pairs suitable for the model.

- **Input Format**: The input includes the sentence, the ambiguous pronoun, and the candidate antecedent, structured in a natural language prompt for resolution.
- **Output Format**: The output is a binary label, `"true"` or `"false"`, representing whether the candidate resolves the pronoun.

Original data:

{

  "text": "The trophy doesn't fit in the suitcase because it is too small.",

  "span2_text": "it",

  "span1_text": "trophy",

  "label": 0

}

Processed input:

resolve: The trophy doesn't fit in the suitcase because it is too small.

Pronoun: it

Candidate: trophy

Processed output:

False

## TOKENIZATION

Tokenization is a fundamental step in T5 (Text-to-Text Transfer Transformer) that involves converting raw text into a sequence of tokens—units the model can process. T5 employs the **SentencePiece tokenizer**, which uses a subword-based approach, allowing for efficient handling of rare words and unknown tokens.

**Input Text**:
*"Tokenization is essential for NLP tasks."*

**Tokenization:**

```
['_Core', 'ference', '_resolution', ':', '_Mark', '_told', '_Pete',
'_many', '_lies', '_about', '_himself', ',', '_which', '_Pete',
'_included', '_in', '_his', '_book', '.', '_He', '_should', '_have',
'_been', '_more', '_skeptical', '.', '_Does', '_', "'", 'He', "'",
'_refer', '_to', '_', "'", 'Mark', "'", '?']
```

**Token IDs**:
```
[9020, 11788, 3161, 10, 2185, 1219, 19786, 186, 7797, 81, 2448, 6, 84,
19786, 1285, 16, 112, 484, 5, 216, 225, 43, 118, 72, 27716, 5, 3520,
3, 31, 3845, 31, 2401, 12, 3, 31, 19762, 31, 58]
```

## TRAINING

### Training Hyperparameters

Hyperparameters are configuration settings that govern the training process of a model, such as learning rate, batch size, number of layers, and dropout. They influence how the model learns from data, affecting its speed, stability, and overall performance. Several hyperparameters were varied during training to assess their impact:

- **Learning Rate**:
  - Learning rate controls how much the model's parameters are adjusted during each update.
- **Batch Size**:
  - Batch size determines how many examples are used per training step.
  - Smaller batch sizes (e.g., 4) allow more granular updates but can introduce noisy gradients.
  - Larger batch sizes (e.g., 16) stabilize training but require more memory and might converge more slowly.
- **Number of Layers**:
  - The number of encoder and decoder layers affects model depth and capacity.

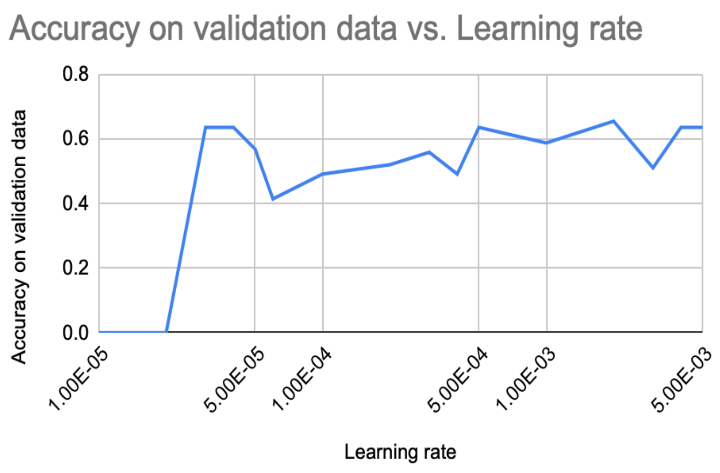
    - Experiments were conducted with 4, 6, 8, and 12 layers.
- **Dropout**:
    - Dropout  is a regularization technique that randomly disables neurons during training to prevent overfitting.
- **Epochs**:
    - The model was trained for a fixed number of epochs with **early stopping** based on validation accuracy.

**Training Process**

1. **Loss Function**:The loss function is **cross-entropy loss**, which measures the difference between predicted and true antecedents.
2. **Optimizer**: **AdamW** (Adam optimizer with weight decay) was used for optimization, as it handles sparse gradients effectively and prevents overfitting.
3. **Gradient Accumulation**: For smaller batch sizes, gradient accumulation was employed to simulate larger batches and stabilize training.
4. **Scheduler**: A learning rate scheduler was used to decrease the learning rate over time, helping the model converge to a better solution.
5. **Validation**: After each epoch, the model's performance was evaluated on the validation set. The accuracy metric was used to measure how many pronouns were correctly resolved.

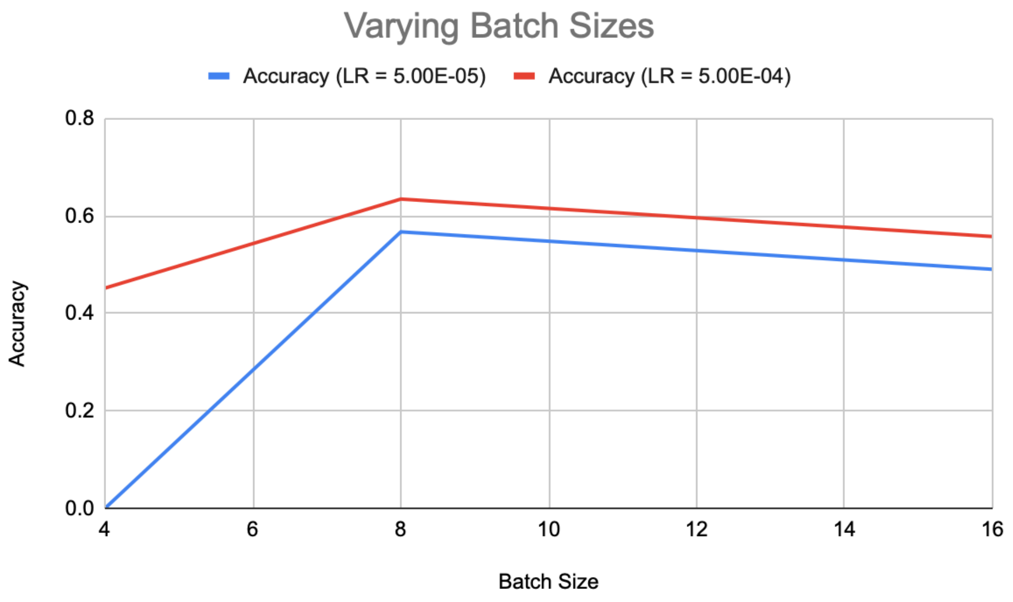
## RESULTS

### 1.Learning Rate



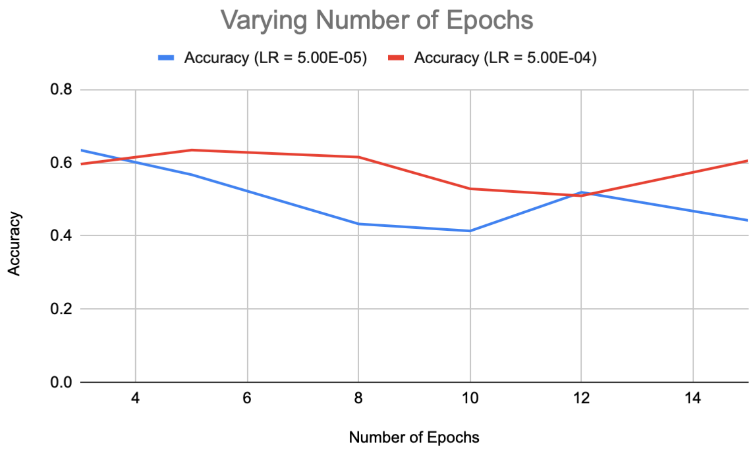
The results highlight the importance of careful tuning. While low rates are stable but ineffective, excessively high rates cause instability. The most reliable performance was observed around 5.00E-04, which is why we have chosen  it  for future training configurations.

**2.Batch Size**



The results show that batch size importantly effects performance and it especially influences outcomes when paired with different learning rates. When the learning rate is set low at 5.00E-05, smaller batches with a size of 4 do not learn effectively, but a batch size of 8 manages to achieve the best accuracy of 0.5673 because it strikes a balance between gradient stability and the frequency of updates.  Larger batches (size 16) smooth gradients too much, slowing convergence. For the higher learning rate (5.00E-04), batch size 8 again performs best (0.6346), as it benefits from stable  updates. In contrast, batch size 4 is too unstable, and batch size 16 converges suboptimally, highlighting the importance of matching batch size with learning rate.

## 3. Epochs



**Learning Rate 5.00E–05:**

The accuracy starts high at 3 epochs (0.6346), indicating quick improvement during initial training. However, accuracy steadily decreases as the number of epochs increases, reaching a low of 0.4135 at 10 epochs.  Some recovery occurs by 12 epochs (0.5192), but it drops again at 15 epochs (0.4423)

Trend: Possible overfitting or saturation occurs, as the model performance degrades with extended training.

**Learning Rate 5.00E–04:**

The accuracy starts lower than the smaller learning rate at 3 epochs (0.5962), but improves and peaks at 5 epochs (0.6346). After 5 epochs, accuracy begins to decline, with some fluctuations. At 15 epochs, accuracy improves slightly (0.6058), suggesting some stabilization.

Trend: The model benefits from a slightly faster learning rate initially but also struggles with prolonged training.

## 4. Number of Layers - Encoder and Decoder

| #Layers(Encoder and Decoder) | Accuracy on validation data | Learning rate |
|---:|---:|---:|
| 4 | 0.4615 | 5.00E-04 |
| 6 | 0.6346 | 5.00E-04 |
| 8 | 0.5385 | 5.00E-04 |
| 8 | 0.5673 | 1.00E-05 |
| 12 | 0 | 1.00E-05 |

For 6 and 8 layers, the learning rate of 5.00E–04 generally results in better performance than 1.00E–05, except for slight improvement in the 8-layer model (0.5673) at 1.00E–05. At 1.00E–05 and higher layer counts (12 layers), the model fails to train effectively, as indicated by 0 accuracy.
Increasing the number of layers improves performance up to 6 layers but leads to diminishing returns and instability as the layers increase to 8 or 12.

## 5. Scheduler

| Scheduler | Accuracy on validation data ( in %) |
|---|---|
| cosine with decay | 50.96 |
| polynomial | 50 |
| constant | 61.54 |
| reduce lr on plateau | 57.69 |

Constant Scheduler:

- The constant learning rate achieves the highest accuracy (61.54%), indicating that the model performs best when the learning rate remains fixed throughout training.

Reduce LR on Plateau:

- This scheduler achieves the second-highest accuracy (57.69%). It adjusts the learning rate dynamically when the model stops improving, which seems to help but doesn't outperform a constant learning rate.

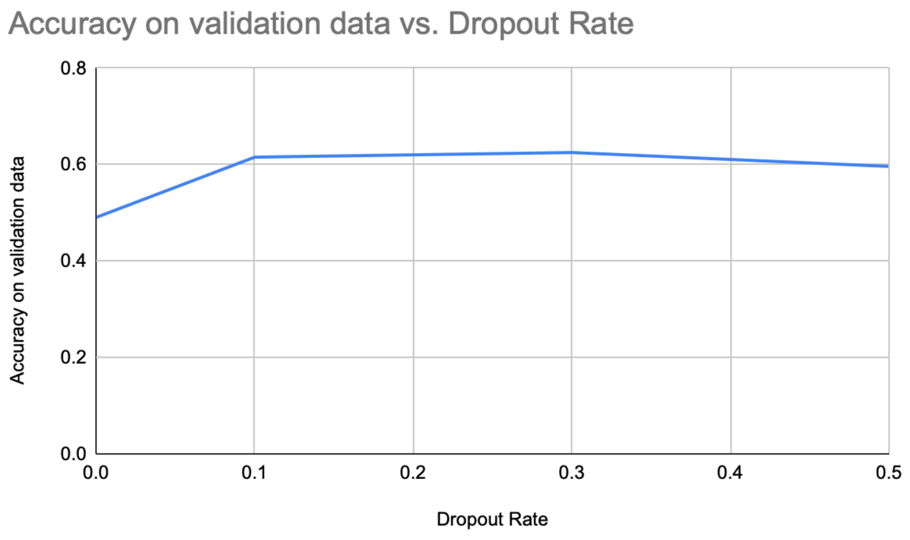
Cosine with Decay:

- The cosine decay scheduler performs worse (50.96%) than both constant and "reduce LR on plateau" strategies. The periodic decay has not aligned well with the training dynamics, leading to suboptimal performance.

Polynomial Scheduler:

- The polynomial scheduler yields the lowest accuracy (50.00%). Its aggressive learning rate decay might cause the model to under-train, especially if the initial learning rate isn't high enough.

**6.Dropout Rate**



The results show that regularization balances with model capacity and both play important roles. Without dropout (0) the model achieves relatively low accuracy (0.4904) and this likely happens due to overfitting because it fully relies on all neurons during training. Dropout at 0.1 improves accuracy importantly—this prevents over-reliance on specific neurons and maintains sufficient model capacity. A dropout rate of 0.3 improves accuracy to 0.625 because additional regularization helps the model generalize better.

Increasing dropout to 0.5 slightly reduces accuracy to 0.5962 and the model begins to lose too much information which limits its ability to learn complicated patterns. Moderate dropout rates (0.1–0.3) prevent overfitting and retain enough capacity for effective learning.

## CONCLUSION

All the above Hyper-parameter tunings shown above display how the model is affected whenever one or more of the hyper-parameters is changed.
The best result that was achieved was an accuracy of 65.38% (0.6538) when the learning rate was increased to 2.00E-03 while the batch size, number of epochs and everything else was kept unchanged.

Further, no class imbalance was found in the dataset for WSC, with no class imbalance, 53% was False 47% was True.

Potential ways to further fine-tune the model:

- One can try parameter efficient fine-tuning (PEFT) using approaches like LoRA (low-rank adapters) to tune only a small subset of model parameters for faster training.
  This might be useful considering the dataset for WSC on SuperGLUE is small.
- Experiment with freezing the initial layers and only fine-tuning the outer layers.

## REFERENCES

1. **Raffel et al. (2020)**: *Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer*
 arXiv:1910.10683
 This paper introduces the T5 model, detailing its architecture and pre training objectives, which are fundamental for NLP tasks, including coreference resolution.

2. **Wang et al. (2019)**: *SuperGLUE: A Stickier Benchmark for General-Purpose Language Understanding Systems*
 SuperGLUE Paper
 SuperGLUE is a benchmark suite for evaluating NLP models, including tasks like coreference resolution, which are useful for assessing T5's performance on complex language tasks.

3. **Baddage, K. (2021)**: *Introduction to Coreference Resolution in NLP*
 Medium Article
 This article provides an approachable introduction to coreference resolution, explaining its significance and applications in NLP.