FORM UNDERSTANDING USING SUPERVISED FINE TUNING MACHINE LEARNING ALGORITHM ON A PRETRAINED LARGE LANGUAGE AND VISION MODEL

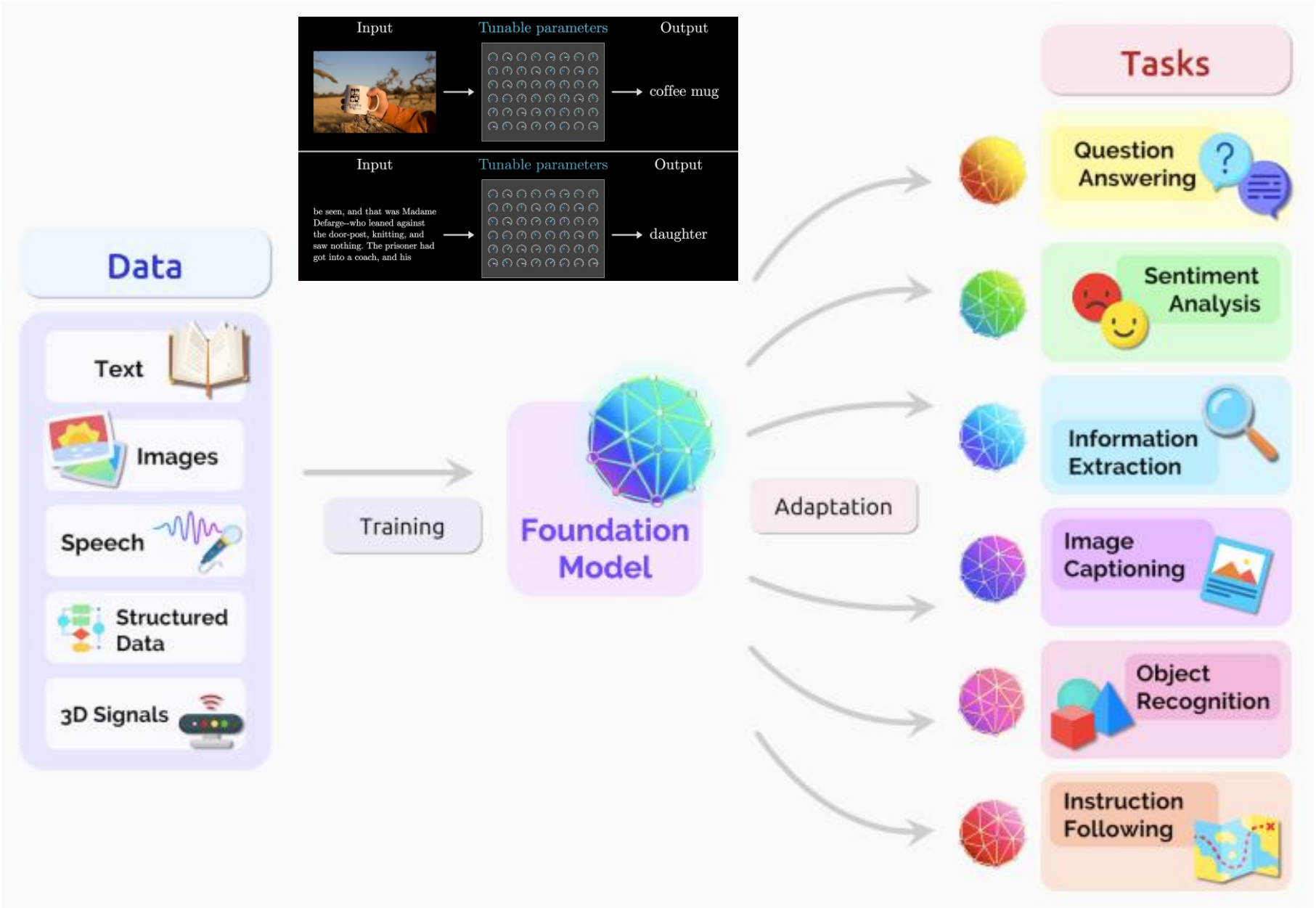
Arpit Agrawal

Thesis Presentation

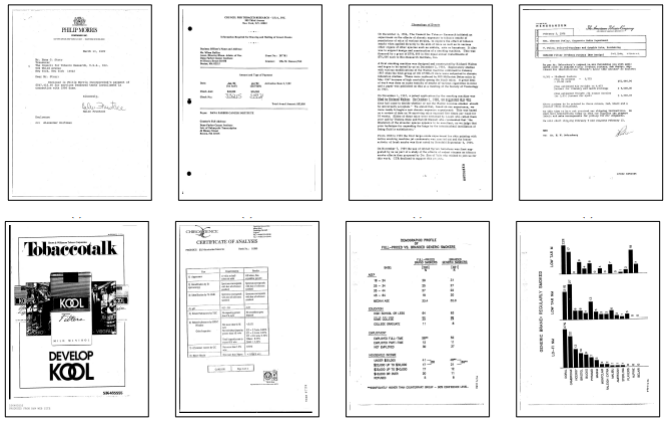
April 2024

Pre-trained Large Language Model (LLM)

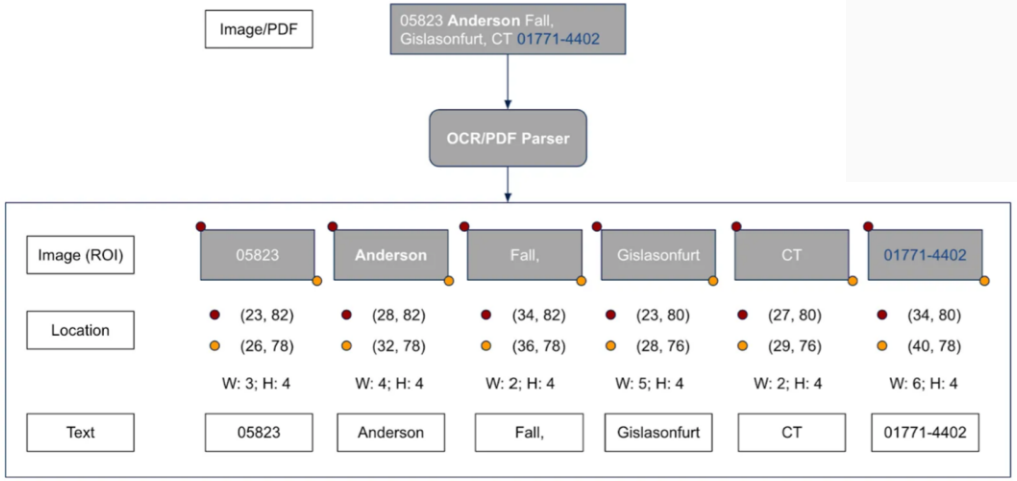
- Large dataset(100s GBs or TBs)
- Unsupervised Training
- Model Learns Context
- With language input it learns grammar rules, linguistic patterns, factual information, and reasoning abilities
- However without adaptation, accuracy of the model on a certain task is quite low

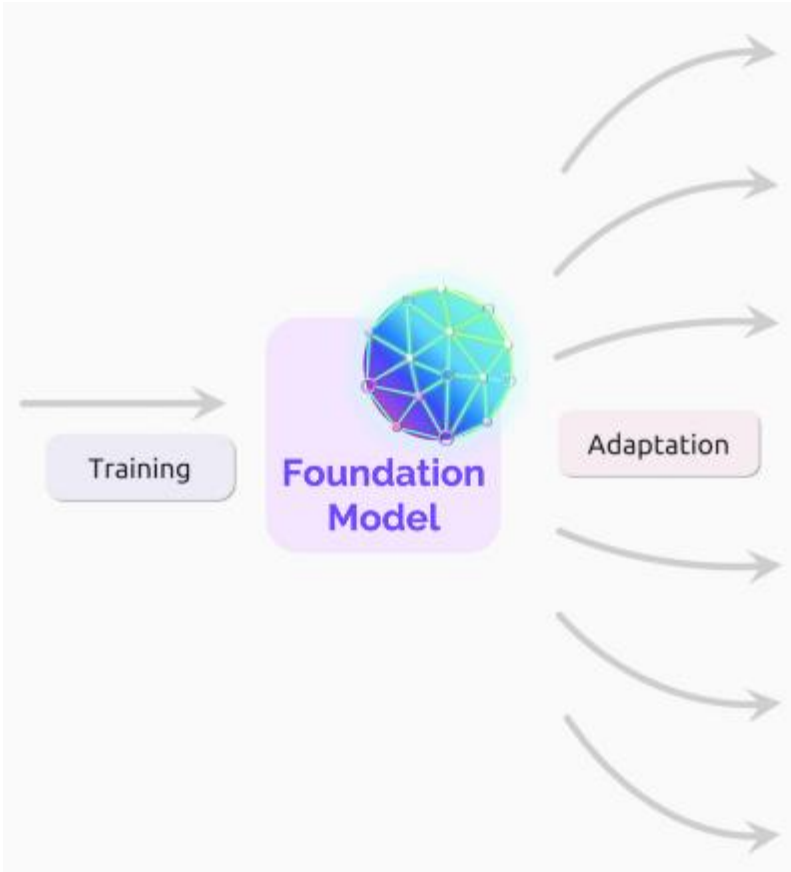


Pre-trained LLM for Document Understanding



- OCR Text
- OCR Text + Bounding Box
- OCR Text + Bounding Box + Image Feature map(CNN)
- OCR Text + Bounding Box + Image(ViT)





Document Layout Analysis

Document Parsing

Document Image Processing

Document Table Detection

Table Structure Recognition

Reading Order Prediction

Document Visual Question Answering

Document Information Retrieval

Document Key Information Extraction

Document Classification and Categorization

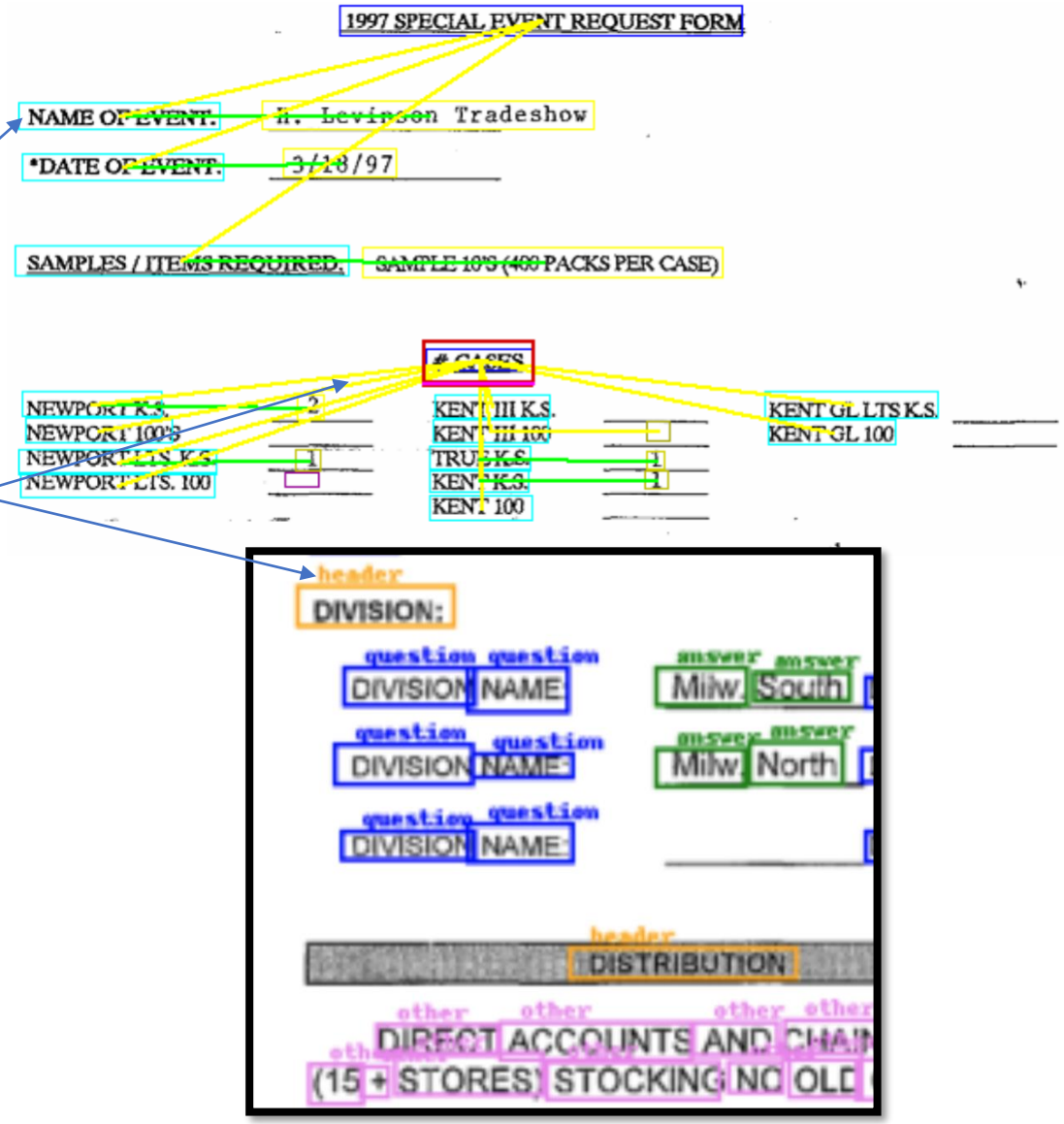
Historial Document Content Understanding

Table-based Question Answering

Fine-tuning Pre-trained LLM for Form Understanding

Fine-tuning

Word grouping
Semantic Entity Labelling
Semantic Entity Linking



- ❑ Full fine-tuning approaches
- ❑ Parameter-efficient fine-tuning methods (PEFT)
- ❑ Prompt engineering strategies
- ❑ Multi-task learning
- ❑ Adapter-based fine-tuning
- ❑ Meta-Adapters: Parameter Efficient Few-shot Fine-tuning
- ❑ Sandboxed tuning environments

Aim and Objectives

The primary aim of this research is to implement a fine-tuning technique for a pre-trained model to achieve state-of-the-art performance in form information extraction task

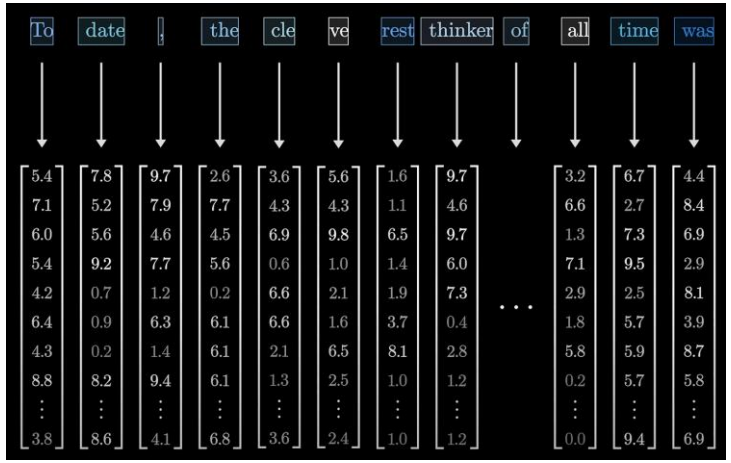
The research objectives are defined in line with the purpose of this study and are listed below:

- To fine-tune a pretrained large language model targeting form information extraction
- To evaluate the fine-tuned model performance on benchmark dataset

Challenge : How best to incorporate spatial and layout information

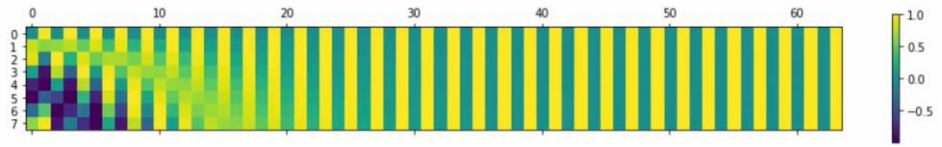
- pre-processing layout information
- using CNNs to develop feature maps of the high-level spatial information
- adding it at different levels (early fusion or late fusion) in the transformer architecture (LayoutLM vs LMv2)
- using different pre-training objectives to learn the spatial context
- formulate correct reading order before feeding the positional information to the model

Text | Layout | Image

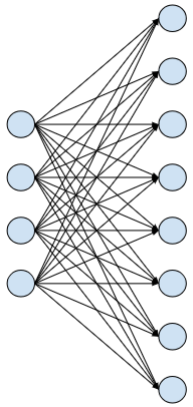


Text -> Tokens -> Vectors

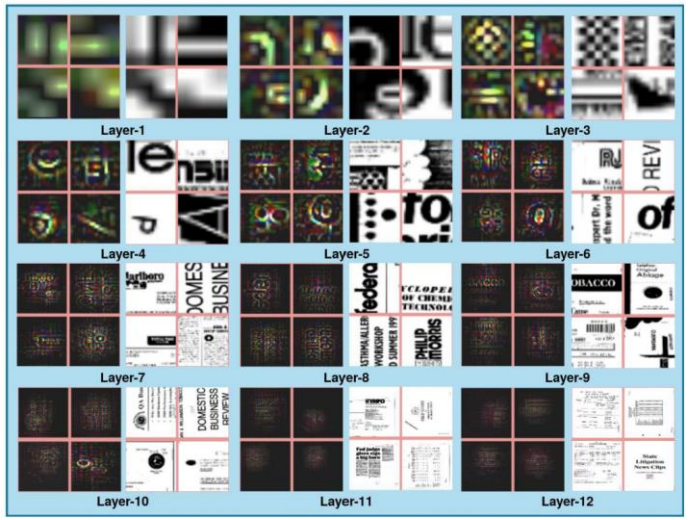
Each Model has its own Vocabulary
Its own tokenization encoding
Its own token ids



Sinusoidal encodings



Using a linear projection layer to map bounding box coordinates to model dimension



The initial layers of a CNN network learn more **generic features** while higher layers learn class **specific features**

Generic features : **horizontal edges, vertical edges, diagonal edges, circular edges**, corresponding to circular parts of English characters. Intermediate Layers learns **small structure** presents in document type such as characters (like ‘e’, ‘5’, ‘p’ and ‘A’) and small circular structures. Final Layer learns **features corresponding to text** present in a complete document.

Literature Review

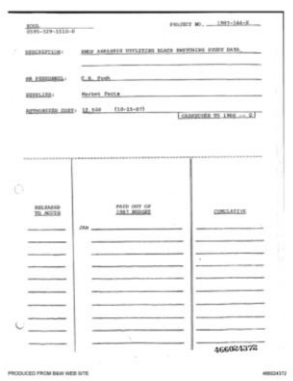
MODEL	MODEL TYPE	DATA MODALITIES	FEATURES
LAMBERT	Encoder	Text, Layout	Language modeling, attention mechanisms, document layout focus
BERTGrid	Encoder	Text, Layout	BERT extension for grid-like structures, form and table processing
LayoutLM	Encoder	Text, Layout, Visual	Text, layout, visual integration for document understanding(late fusion)
LayoutLMv2	Encoder	Text, Layout, Visual	Enhanced LayoutLM(early fusion), refined multimodal document processing capabilities
TILT	Encoder-Decoder	Text, Layout, Visual	Transformer encode-decode, layout and visual information handling for translation or summarization
BROS	Encoder	Text, Layout	Sentence representation with layout awareness
LayoutT5, LayoutBART	Encoder-Decoder	Text, Layout, Visual	Adaptation of T5 and BART for document understanding with text, layout, and visual features
DocFormer	Encoder	Text, Layout, Visual	Transformer-based, processes text, layout, and visual elements for document interpretation

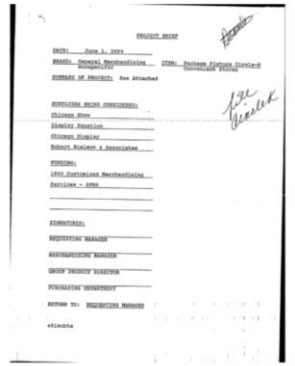
Exploratory Data Analysis – FUNSD Dataset

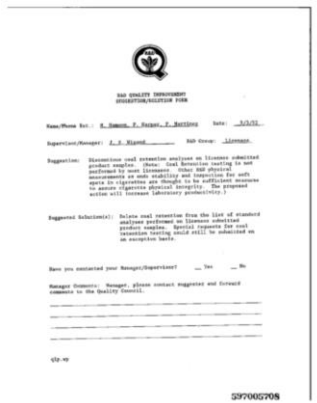
Subset	No of Forms	No of Words	No of Entities	No of Relations
Training	149	22512	7411	4236
Testing	50	8973	2332	1076

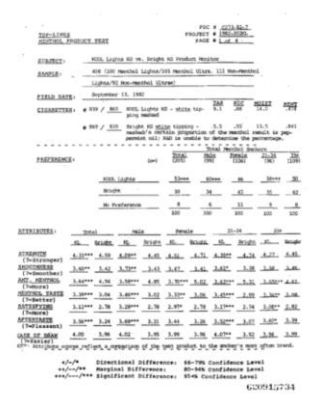
Subset	Header	Question	Answer	Other	Total
Training	441	3266	2802	902	7411
Testing	122	1077	821	312	2332

- 199 Document images with diverse formats
- 100 dpi resolution
- Noisy due to repeated scanning and printing
- 7411 Entities can be used for training on Entity Labelling Task
- 4236 Relations can be used for training on Entity Linking Task
- Adequate for Fine-tuning a Pre-trained a Large Language Model



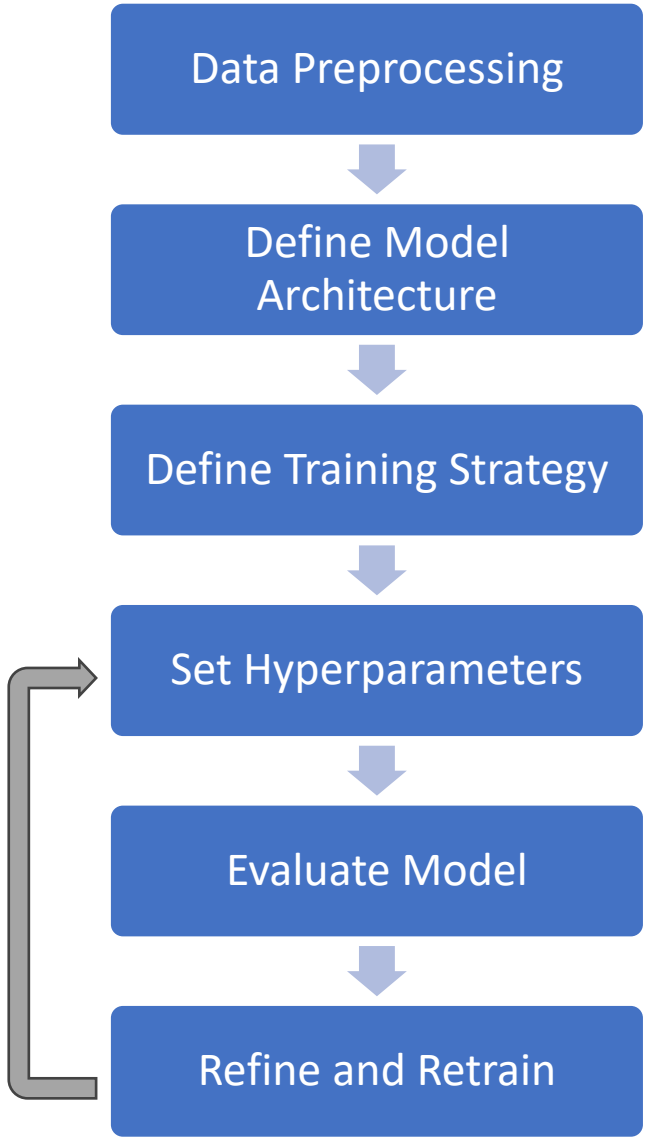






Methodology

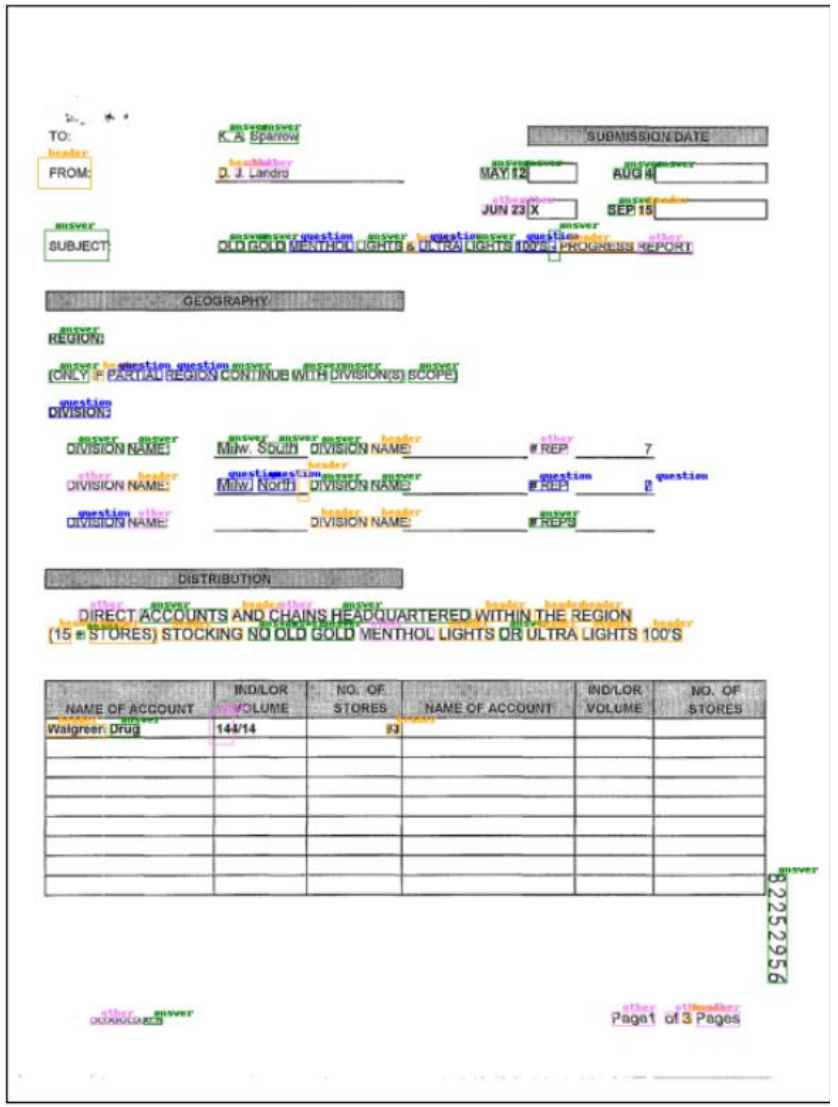
1. **Data Preparation**
   - Acquire and format the FUNSD dataset for LayoutLMv2 compatibility.
   - Pre-process for token-level annotations from world level annotations.
2. **Model Architecture**
   - Utilize pre-trained LayoutLMv2 for understanding document layout and text.
   - Fine-tune using Hugging Face transformers library on google Colab
3. **Training Strategy**
   - Employ cross-entropy loss for optimization (penalize wrong prediction more)
   - Partition FUNSD dataset for training and testing.
   - Use AdamW optimizer for better model parameter updates and fine-tuning.
4. **Hyperparameter**
   - Set learning rate, batch size, and epochs.
5. **Evaluation**
   - Test against benchmarks using Accuracy, Precision, Recall, and F1-score using FUNSD Test set.
   - Focus on achieving high scores to ensure comprehensive document understanding.
7. **Feedback and Iteration**
   - Refine and retrain based on feedback for optimal model performance.



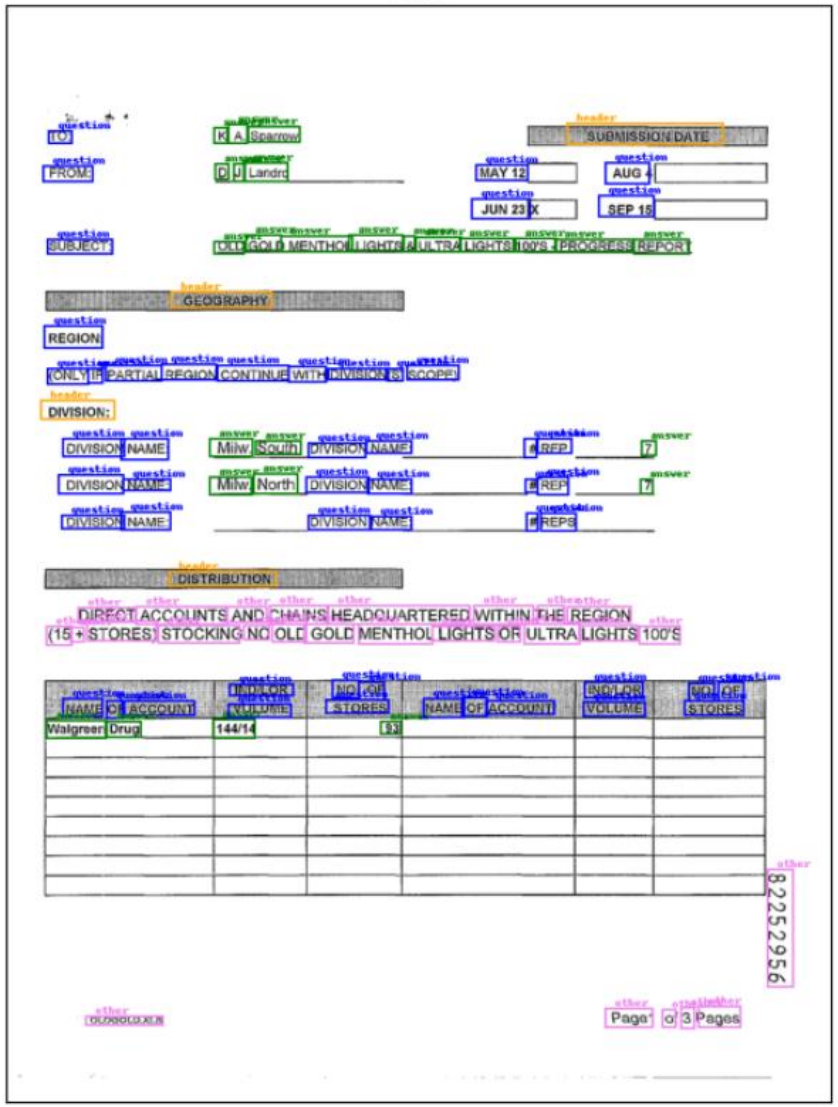
Results and Discussion – Inference from Pre-trained Model

- For pre-trained model label(ids) hold no meaning
- Fails to recognize contextual entity relationships.
- Labels non-text elements erroneously (e.g. “=“).
- Does not differentiate between similar entity categories.
- Struggles with multi-word entities, inconsistently labels parts.
- Mislabels data due to lack of domain-specific training.

Prediction



True



Results and Discussion – Inference from Fine-tuned Model

What the Model got Right:

- It **correctly identified** and **classified** several headers, questions, and answers with **precise bounding boxes**.

- The model was able to **distinguish between different entity types** for many entities.

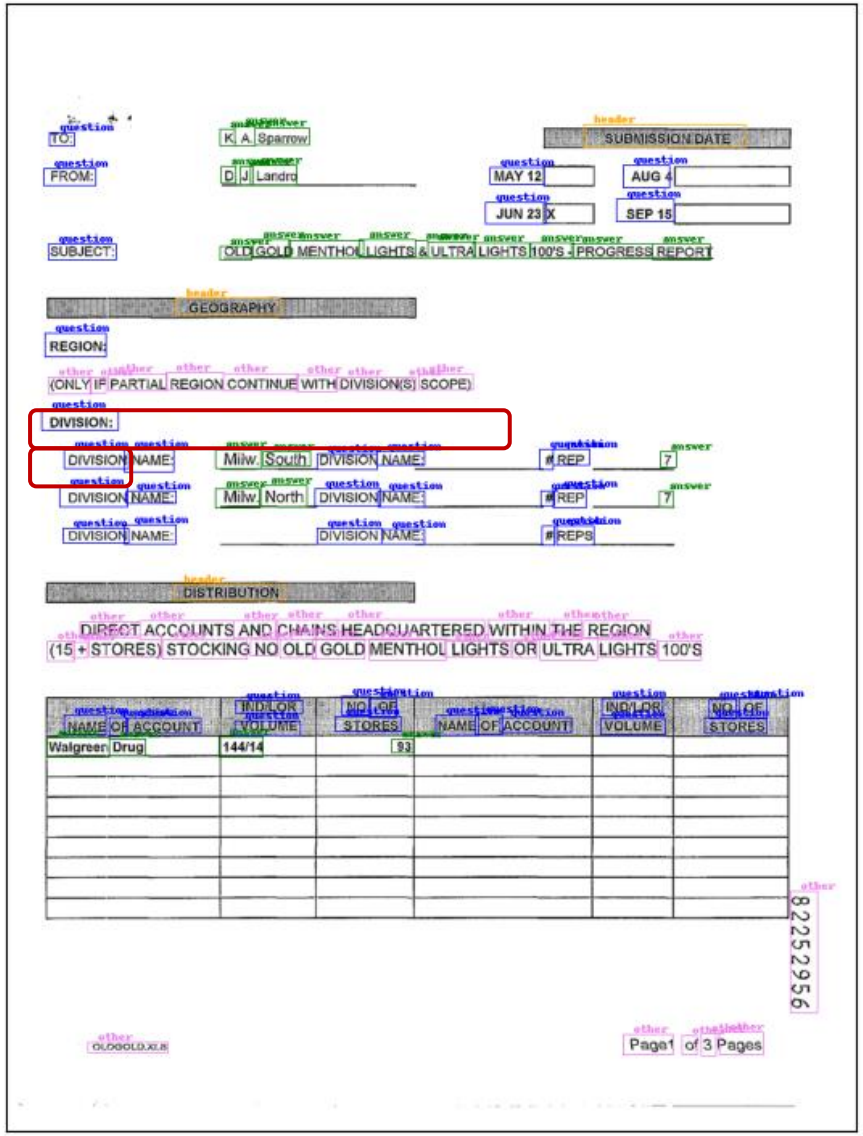
What the Model got Wrong:

- There are instances where the model has **misclassified certain entities**, for example, some entities labelled as 'other' might be 'question' or 'answer'.

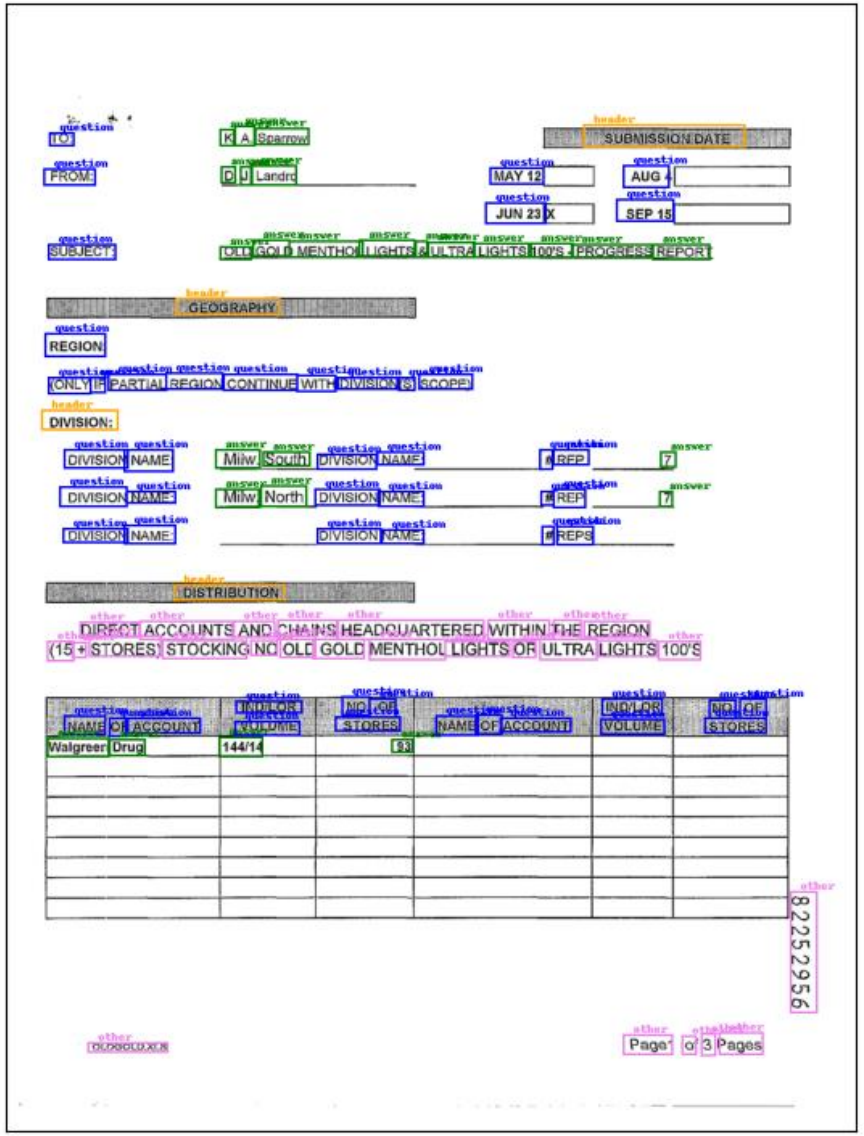
- There are discrepancies in the number of entities identified, where the model either missed some entities or incorrectly segmented and classified single entities into multiple parts.

	ANSWER	HEADER	QUESTION	overall_precision	overall_recall	overall_f1	overall_accuracy
precision	0.760405	0.572816	0.820072	0.78265	0.823884	0.802738	0.811931
recall	0.835600	0.495798	0.851643	0.78265	0.823884	0.802738	0.811931
f1	0.796231	0.531532	0.835560	0.78265	0.823884	0.802738	0.811931
number	809.000000	119.000000	1065.000000	0.78265	0.823884	0.802738	0.811931

Prediction



True



Hyperparameter Tuning

Learning Rate

- **Best performance** -> 5.00E-05 -> F1 Score 0.802
- **Highest Recall** -> 0.823 -> strong ability to identify all relevant instances.
- **Worst performance** -> 6.00E-05 -> F1 Score 0.76
- **Higher learning rates (up to a point) may help the model better optimize its weights and biases**.
- Beyond this optimal point (5.00E-05) model beginning to overshoot the optimal weights, thus reducing its ability to generalize the data accurately.

Learning Rate	Precision	Recall	F1	Accuracy
1.00E-05	0.771	0.811	0.79	0.809
2.00E-05	**0.799**	0.797	**0.798**	0.8
3.00E-05	0.761	0.789	0.775	**0.814**
4.00E-05	0.742	0.799	0.769	0.81
5.00E-05	0.783	**0.823**	**0.802**	**0.812**
6.00E-05	0.744	0.777	0.76	0.765

Number of Epochs

- **Epoch 1 ->** All metrics < 0.5 -> **model is just beginning to learn** from the data.
- **Epoch 5 ->** All metrics > 0.75 -> **model has learned a substantial amount from the training data**.
- **Epoch 10 -> Peak performance -> F1 score 0.802 Accuracy 0.812. Model balance between overfitting and underfitting is optimal**.
- **Epoch > 10 ->** noticeable drop in all metrics Precision 0.747 F1 0.776.

Epochs	Precision	Recall	F1	Accuracy
1	0.466	0.36	0.406	0.451
5	0.775	0.792	0.783	0.799
10	**0.783**	**0.823**	**0.802**	**0.812**
15	0.761	0.798	0.779	0.802
20	0.771	0.816	0.792	0.786
25	0.747	0.808	0.776	0.789

Hyperparameter Tuning

Batch Size

- **Size 2** -> lowest recall and F1 score
- **Size 4** -> highest F1 score and accuracy of all batch sizes tested, improves recall significantly without compromising much on precision
- **Size 6** -> decrease in all metrics, suggesting it might not be as efficient as smaller batches for this dataset and model configuration
- **Size 10** -> slightly better than for batch size 2 but still lower than for batch size 4

Batch Size	Precision	Recall	F1	Accuracy
2	0.784	0.765	0.774	0.776
4	**0.783**	**0.824**	**0.803**	**0.812**
6	0.752	0.791	0.771	0.783
10	0.782	0.796	0.789	0.81

Overall, a **batch size of 4 seems to provide the best balance** between precision, recall, F1 score, and accuracy for this dataset and model.

Interpretation

Generalization: Batch size 4 seems to strike a balance between these extremes. It is large enough to provide stability in the gradients, reducing the noise seen in very small batches like 2, yet small enough to retain some level of noise, which helps in escaping poor local minima and promotes better generalization.

Conclusion

- We began with a thorough **exploratory data analysis** that informed our preprocessing and model configuration strategies.
- The preprocessing steps were designed to transform image data and annotations into a model-friendly format, laying the groundwork for effective model training.
- The we carried out fine tuning - **F1 score of 0.802 and an Accuracy of 0.812 –** model able to generalize the knowledge it had learned without overfitting to the training data.
- Adjustments to the learning rate, batch size, and epochs were made with the goal of optimizing the model's performance, with **5e-5 learning rate, 10 epochs and a batch size of 4** emerging as the most effective configuration. **Higher learning rates (up to a point) may help the model better optimize its weights and biases.**
- Thus post required preprocessing, model selection, and strategic fine-tuning, a pre-trained LLM can be adapted for the task of form understanding.

Future Recommendation

- **Datasets** : Larger refined datasets can be used for fine-tuning. Fine-tuning on invoice dataset could significantly enhance its performance on forms due to shared crucial tasks such as word grouping, entity labelling, and entity linking. Invoices, like forms, consist of structured data where entities like dates, amounts, and names are critical for comprehension.
- **Layout Embeddings** : Different methods of encoding the bounding box coordinates. Relative positional, sinusoidal, etc to be explored. Still an active area of research
- **Fine-tuning methods/ensemble** : LORA, QLORA, leveraging Axolotl library for Adapter fine tuning, etc
- **Pre-trained Model Selection** : Decoder, Encoder-Decoder model performance on form understanding. A comparative study on model sizes will also help understand effect of model size on performance. Hardware requirements to be scaled up proportionally.
- **Document Image Pre-processing/Augmentation** : To develop feature maps that emphasize linked entities and are able to separately encode general text versus labelled entities

Thank you