**Image Document classification using layoutlmv2**

Layoutlmv2 is used for the document classification because it is specifically designed for handling document layout information, making it a suitable choice for document classification tasks. The model is designed to handle document layout information along with textual content, making it suitable for tasks like document classification. I implemented functions for data preprocessing (e.g., tokenization and encoding), training the model using a DataLoader, and validating the model on a separate validation dataset.

**Requirements:**

* Python 3.6 or later
* Transformers library (for LayoutLMv2)
* PyTorch library
* Hugging Face datasets library
* PIL (Python Imaging Library) for image processing
* tqdm (optional) for progress tracking during training

**Installation:**

1. clone the repository

*git clone https://github.com/keskhanal/layoutlmv2-document-classification.git*

*cd layoutlmv2-document-classification*

1. Install the required libraries and run the notebook file

**Results:**

while training model for just 5 epochs we are able to achieve almost 88% accuracy in training data and about 86% in validation data. This can be further improved by training more epochs

**Acknowledgments:**

This project is based on the LayoutLMv2 model and transformers library from Hugging Face.

**Challenges:**

* **Data preprocessing:** Dealing with a combination of images and textual data can be challenging. Ensuring that the images and texts are properly aligned and tokenized can be complex, especially if the data comes from various sources with different layouts.
* **Model training:** Fine-tuning a complex model like LayoutLMv2 requires sufficient computational resources and time. Training deep learning models on large datasets can be computationally intensive.
* **Hyperparameter tuning:** Choosing the right learning rate, batch size, and other hyperparameters can significantly impact the model's performance. Finding the optimal hyperparameters might require extensive experimentation.

**Model performance can be improved by using following techniques:**

* **Data augmentation:** Expanding the dataset through data augmentation techniques for images and text can help improve the model's generalization.
* **Experiment with different architectures:** You might try variations of the LayoutLMv2 model or experiment with other transformer-based models for document classification to see which one performs best for your specific task.
* **Hyperparameter search:** Spend more time fine-tuning hyperparameters to optimize the model's performance.
* **Error analysis:** Analyze the model's misclassifications to identify patterns and potential areas of improvement. This can help you gain insights into the model's strengths and weaknesses.