# Modèles de Langage de Grande Taille (LLMs) et Applications Avancées

**TP : Pré-entraînement et Fine-Tuning d'un Modèle de Langage**

## Partie 2 : Fine-Tuning avec BERT

Utiliser **BERT pré-entraîné** pour une tâche de classification binaire (ex. avis positifs/négatifs).

1. **Dataset :**
   - Fournir un dataset simple ou utiliser le dataset **IMDB** (pour la classification des avis disponible dans torchtext).
   - Exemple de données :
     - "J'adore ce film !" → Positif (1)
     - "Ce film est horrible." → Négatif (0).
2. **Préparer le dataset**
   - Diviser les donnees en jeu d'entraînement et jeu de validation (train_test_split) de sklearn
   - Tokeniser (`BertTokenizer.from_pretrained` (de `transformers`))
3. **Chargement du modèle pré-entraîné :**
   - Charger un modèle BERT prêt à être ajusté pour une tâche de classification.
   - `BertForSequenceClassification.from_pretrained` (de `transformers`) : Pour charger BERT et ajouter une couche dense pour 2 classes.

4. **Adapter BERT à la classification :**
   - Convertir les données tokenisées en datasets compatibles avec PyTorch. `TensorDataset` (de `torch.utils.data`) : Pour transformer les encodings et labels en datasets.

5. Configurer l'entrainement
   - Définir les paramètres d'entraînement (nombre d'époques, batch size, etc.). `TrainingArguments` (de `transformers`) : Pour définir les paramètres d'entraînement.

6. **Entraîner le modèle :**
   - Utiliser les données d'entraînement pour ajuster les poids de BERT et de la nouvelle couche.

7. **Tester le modèle**