AG News Classification with Transformer and DNN Architectures

1. Project Overview

This project investigates AG News classification using two neural architectures: - A baseline **DNN classifier** - A custom **Transformer-based classifier**

We aim to explore differences in architecture behavior, input handling, and performance under MILA's evaluation style, with a focus on generalization, sequence handling, and training dynamics.

---

2. Dataset

- **AG News**: News headlines + descriptions
- **Task**: 4-class classification (World, Sports, Business, Sci/Tech)
- **Inputs**: Headline + Description (concatenated)
- **Size**: ~120k training samples
- **Tokenization**: Word-level regex tokenization (`\b\w+\b`)
- **Padding**:
  - DNN: static padding (`max_seq_len = 100`)
  - Transformer: dynamic padding per batch with `pad_sequence`

---

3. Model Architectures

3.1 DNNClassifier (Baseline)

- **Embedding → MeanPool → FC1 → Dropout → FC2 → Logits**
- `Dropout=0.3`, no batch norm (BN hurt validation)
- F1: ~91% macro

3.2 TransformerClassifier (Scratch)

- Learned positional encodings
- 2 encoder layers, 4 heads each
- Mean pooling with masking (not using [CLS])
- Masked mean pooling handles `<pad>` tokens
- Achieved ~97.4% macro F1 on validation

---

4. Data Pipeline

- Built flexible `TextDataset`:
  - Supports static and dynamic padding

- Switches logic based on `model_type` and config
- Implemented `transformer_collate_fn(pad_idx)`:
  - Uses `pad_sequence` for dynamic batch padding
  - Returns `src_key_padding_mask` to mask attention on padded tokens
- Config-driven design (`config.yaml`)

---

5. Transformer Forward Pass (Explained)

- Input: `[batch_size, seq_len]` token IDs
- Embedding + Positional Encoding: `[B, L, D]`
- TransformerEncoder (2 layers $\times$ 4 heads)
- Masked Mean Pooling:
  - Uses `src_key_padding_mask`
  - Avoids influence of `<pad>` tokens
- Final classifier: Linear projection `[D → num_classes]`

---

6. Attention Explained (Heads, Masking, and Pooling)

- 4 attention heads look at different token relationships
- Each token attends to others based on learned relevance (scaled dot product)
- Head outputs are concatenated → `[B, L, D]`
- Projected using `Linear(D, D)` to blend head info
- Masked mean pooling aggregates final sequence embeddings robustly

---

7. Training Setup

- Optimizer: Adam
- Scheduler: ReduceLROnPlateau (monitoring macro F1)
- Gradient clipping: `clip_grad_norm_`
- Early stopping based on F1 (patience = 5)
- Stratified validation split (10%)
- Static vs. dynamic padding evaluated

---

8. Key Observations

- Transformer outperformed DNN by 6–7% F1, even without pretraining
- Mean pooling with masking was essential
- Validation sometimes higher than training → possibly due to regularization effects

- Training loss may not reflect generalization performance
- F1 score chosen as main metric due to class imbalance and MILA recommendation

---

9. Future Work

- Hands-on exploration of Q/K/V tensors per head
- Deeper look at attention maps and interpretability
- Extend to pretrained models (e.g., T5, BERT, Flan-T5)
- Multi-task setup: headline-only vs headline+desc
- Evaluate Byte-BPE vs word-level for robustness

---

10. Appendix

- Full config file
- Selected training logs
- Visualizations of attention and pooling
- Key implementation notes (collate, masking logic, etc.)