**MSO 3255 Assessment 2: Mini Project**

**Project Title: Improving and Evaluating a Distilled Question Answering Model**

**Instructions:**

**1. Introduction**

In this project, you will start with an **end-to-end PyTorch codebase** that demonstrates knowledge distillation on **SQuAD v1.0**. The baseline code uses a **fine-tuned BERT large** as the teacher model and a **smaller BERT** as the student model. Your goal is to **enhance** this baseline in meaningful ways, thereby learning about advanced techniques in model distillation, data preprocessing, hyperparameter tuning, and evaluation for QA.

**2. Project Objectives**

1. **Review & Run the Baseline**
   * Understand how data loading, teacher inference, distillation training, and QA evaluation are implemented in the provided code.
2. **Enhance the Baseline Model**
   * Implement **4** improvements (see Section 5 below) to increase the student model’s performance, reduce its size, or improve inference speed.
3. **Compare & Report**
   * Measure how each modification affects **Exact Match (EM)**, **F1**, model size, and training/inference speed.
   * Provide a written **analysis** of your findings, supported by tables, plots, or both.

**3. Dataset & Codebase**

1. **Dataset**: SQuAD v1.0
   * A large-scale question answering dataset with human-annotated questions and answers.
   * The provided code uses a **naive approach** (no sliding window) to handle input sequences of length up to 384 tokens.
2. **Baseline Code**
   * A PyTorch script demonstrating:
     + **Data Preprocessing**: Tokenizing (question, context), identifying answer spans.
     + **Teacher Inference**: Collecting final start/end logits from a BERT-large teacher model.
     + **Student Model Training**: Distilling knowledge by combining a **hard loss** (CE with ground-truth) and **soft loss** (KL divergence with teacher outputs).
     + **Evaluation**: A simplified method for computing **Exact Match (EM)** and **F1** on a subset of the validation set.

**4. Project Requirements**

* **Python 3.x** and **PyTorch**
* **Hugging Face Transformers**
* A **GPU environment** is strongly recommended (e.g., Google Colab, local GPU, or a cloud VM).
* Familiarity with basic NLP concepts and QA metrics (EM, F1).

**5. Suggested Improvements**

You must choose **2–4** of the following enhancements (or propose your own ideas in consultation with the instructor):

1. **Sliding Window / Chunking**
   * Split long contexts into overlapping chunks for improved coverage.
   * Combine predictions across chunks to find the best answer.
2. **Official SQuAD Evaluation**
   * Integrate the official evaluation script to properly handle multiple answers, punctuation, etc.
   * Compare results against the simplified metrics in the baseline.
3. **Intermediate Layer Distillation**
   * Match hidden states at certain teacher/student layers, not just final logits.
   * Observe how it affects training time and final performance.
4. **Hyperparameter Exploration**
   * Vary distillation settings (α, temperature), learning rates, batch sizes, or training epochs.
   * Document how each parameter affects performance and model size.
5. **Architectural Variations**
   * Compare different student configurations (fewer/more layers, different hidden sizes, etc.).
   * Plot performance vs. model size and latency.
6. **Pruning/Quantization**
   * Further compress the student using weight pruning or quantization.
   * Evaluate trade-offs in memory footprint, speed, and accuracy.
7. **On-the-Fly Distillation**
   * Instead of storing all teacher logits first, run teacher inference “on the fly” during student training.
   * Useful for memory-constrained environments.

**6. Milestones & Deliverables**

1. **Milestone 1**: Run the Baseline
   * Confirm you can train the **baseline** student.
   * Record baseline metrics (EM, F1), model size, and any issues encountered.
2. **Milestone 2**: Implement Improvements
   * Integrate the enhancements you have chosen into the baseline code.
   * Keep track of any modifications in a **changelog** or **Git version history**.
3. **Milestone 3**: Comparative Experiments
   * Conduct experiments to evaluate the impact of each improvement.
   * Collect **quantitative results**: EM/F1, model size (MB or # of parameters), training time per epoch, inference latency.
4. **Final Deliverables**
   * **Extended Code**: Your final repository or notebook with all modifications.
     1. **Your code should be well**-commented
     2. Submit the final Python code/notebook alongside the report. Please use labcode.mdx.ac.uk and as a submission , share the link to it
   * **Project Report** covering:
     1. **Methodology**: Which improvements you chose and **why**.
     2. **Results**: Tables/plots comparing baseline vs. improved models.
     3. **Analysis**: Observations on performance, unexpected challenges, or failures.
     4. **Conclusion & Future Work**: Summarize your findings and mention any next steps you would take.
     5. Your report should be between 2000-2500 words (at most 8 pages with 11pt font), excluding references. Appropriate use of lists, tables and figures is recommended

**7. Assessment Criteria**

1. **Correctness & Reproducibility** (30%)
   * Does the code run end-to-end without errors?
   * Can the instructor reproduce your results given the same hardware/environment?
   * Regular commitments in Labcode.
2. **Scope & Depth of Improvements** (30%)
   * How many enhancements were implemented and how well were they integrated?
   * Are they **correctly** implemented and explained?
3. **Experimental Rigor** (20%)
   * Did you systematically measure performance changes?
   * Are results well-organized (e.g., tables, graphs)?
4. **Clarity of Report** (20%)
   * Are the code and methods clearly documented?
   * Does the report present clear findings and analysis?
   * Is the report well-organised and structured?

**8. Timeline**

* **Week 20**:
  + Familiarize yourself with the baseline QA distillation code.
  + Attempt a baseline run on a small portion of SQuAD.
  + Choose your **improvement** focus.
* **Week 21**:
  + Implement the chosen improvements.
  + Address any technical issues (e.g., memory constraints, code bugs).
* **Week 22**:
  + Perform systematic experiments.
  + Gather **quantitative results** and start drafting your report.
* **Week 23**:
  + Finalize improvements, wrap up experiments.
  + Start writing your report.
* **Week 24:**
  + Write your report.
  + Finalize and submit your **extended code** and **project report**.
* **Due Date:** 2 May 2025, 6 pm

**9. Resources**

* **Baseline Code**: Provided in the course repository / instructions.
* **PyTorch Documentation**: https://pytorch.org/docs/stable/index.html
* **Hugging Face Transformers**: <https://github.com/huggingface/transformers>
* **Official SQuAD Repo**: <https://github.com/rajpurkar/SQuAD-explorer>