Yêu cầu nâng cấp hệ thống (Project 4.1)

Mục tiêu chính: Nghiên cứu cách kết hợp xử lý dữ liệu và mô hình để tăng độ chính xác phân loại văn bản.

Bộ dữ liệu: 300k samples từ arXiv abstracts (UniverseTBD/arxiv-abstracts-large)

1) Mục tiêu chất lượng

- **Random Forest:** Baseline vững, tối đa Accuracy và Macro-F1
- **AdaBoost:** Cải thiện Macro-F1 ≥ +5pp so baseline
- **Gradient Boosting:** Vượt hoặc bằng RF baseline, tận dụng Feature Fusion
- **XGBoost:** Accuracy ≥ 87.5%, Macro-F1 tối đa
- **LightGBM:** Accuracy ≥ 88%, Macro-F1 cao + calibration

2) 05 Task nghiên cứu độc lập

Task 1: Random Forest + Data Optimization

- **Công nghệ:** cuML (GPU), scikit-learn, NLTK, SVD dimensionality reduction
- **Xử lý dữ liệu:** stopwords context-aware, rare words filtering, lemmatization, TF-IDF + SVD
- **Tuning:** n_estimators, max_depth, max_features, class_weight, threshold optimization
- **Nghiên cứu:** So sánh cuML vs sklearn, tác động SVD lên accuracy, memory optimization cho 300k samples
- **Kết quả:** metrics.json, confusion matrix, performance comparison

Task 2: AdaBoost Rescue

- **Công nghệ:** scikit-learn AdaBoost, DecisionTree base learners, threshold tuning
- **Xử lý dữ liệu:** TF-IDF (1-2 gram), min_df/max_df optimization, feature selection
- **Tuning:** n_estimators, learning_rate, base learner (stump vs depth=2), SAMME vs SAMME.R
- **Nghiên cứu:** Tác động base learner, algorithm comparison, weak learner optimization
- **Kết quả:** so sánh với RF baseline, algorithm performance analysis

Task 3: GBDT + Feature Fusion

- **Công nghệ:** scikit-learn GradientBoosting, sentence-transformers, feature engineering
- **Xử lý dữ liệu:** TF-IDF + Embeddings + feature phụ (length, digit, upper), normalization
- **Tuning:** learning_rate, n_estimators, subsample, max_depth, early stopping
- **Nghiên cứu:** Hiệu quả feature fusion, embedding vs TF-IDF, dimensionality impact
- **Kết quả:** đánh giá hiệu quả fusion, feature importance analysis

Task 4: XGBoost + GPU Optimization

- **Công nghệ:** XGBoost GPU (tree_method="gpu_hist"), RAPIDS cuML, mixed precision
- **Xử lý dữ liệu:** Embeddings + Fusion, caching strategies, batch processing cho 300k samples
- **Tuning:** eta, max_depth, subsample, colsample_bytree, lambda, alpha, GPU flags
- **Nghiên cứu:** GPU vs CPU performance, memory usage, batch size optimization cho dataset lớn
- **Kết quả:** so sánh CPU vs GPU performance, speed benchmarks

Task 5: LightGBM + Calibration

- **Công nghệ:** LightGBM GPU, Platt scaling, Isotonic regression, probability calibration
- **Xử lý dữ liệu:** Embeddings + Fusion, normalization, probability smoothing
- **Tuning:** num_leaves, learning_rate, feature_fraction, bagging_fraction, calibration
- **Nghiên cứu:** Calibration impact, probability reliability, ensemble methods
- **Kết quả:** đánh giá calibration effect, probability quality metrics

3) Kiến trúc hệ thống

Model-Centric Pipeline:

- Mỗi model sẽ tự quản lý toàn bộ pipeline xử lý của mình, bao gồm các bước: preprocessor → vectorizer → feature_fuser → estimator
  - **Preprocessor**: Tiền xử lý văn bản (loại bỏ stopwords, lemmatization, lọc từ hiếm, v.v.) - **hiển thị trong Step 3 khi chọn model**
  - **Vectorizer**: Chuyển đổi văn bản thành vector đặc trưng (BoW, TF-IDF, Embeddings, SVD, ...) - **hiển thị trong Step 3 khi chọn model**
  - **Feature Fuser**: Kết hợp nhiều loại đặc trưng (ví dụ: concat TF-IDF, Embeddings, các feature phụ như length, digit ratio) - **hiển thị trong Step 3 khi chọn model**
  - **Estimator**: Thuật toán học máy chính (Random Forest, AdaBoost, GBDT, XGBoost, LightGBM, ...) => Mỗi model sẽ định nghĩa rõ ràng chuỗi xử lý này, tự động hóa việc xây dựng, huấn luyện, biến đổi dữ liệu và dự đoán. Điều này giúp dễ dàng so sánh, hoán đổi, hoặc mở rộng các pipeline mà không ảnh hưởng đến các thành phần khác.
- API chuẩn: `build_pipeline()`, `fit_pipeline()`, `transform_pipeline()`, `predict_proba()`

Utility Modules:

- `TextPreprocessor`: logic từ DataLoader (stopwords, rare words, lemmatization)
- `VectorizerProvider`: logic từ text_encoders.py (BoW/TF-IDF/Embeddings/SVD)
- `FeatureFusion`: concat TF-IDF + Embeddings + features phụ
- `CalibrationManager`: Platt/Isotonic calibration

4) Streamlit UI (căn theo app.py hiện có)

Cấu trúc hiện tại: 5-step wizard với sidebar navigation

- **Step 1:** Dataset Selection & Upload (đã có)
- **Step 2:** Column Selection (đã có, bỏ preprocessing)
- **Step 3:** Model Configuration & Preprocessing (cần cập nhật)
- **Step 4:** Training Execution (cần cập nhật)
- **Step 5:** Results Analysis (cần cập nhật)

Cập nhật Step 3 - Model Configuration & Preprocessing:

- Model selector với checkbox (RF, AdaBoost, GBDT, XGBoost, LightGBM)
- **Khi tick chọn model → hiện panel tiền xử lý dữ liệu:**
  - Stopwords, rare words, lemmatization options
  - Vectorizer selection (TF-IDF, Embeddings, Fusion)
  - Feature engineering (length, digit ratio, uppercase ratio)
- Panel cấu hình hyperparameters cho từng model

- Preview pipeline configuration trước khi train

Cập nhật Step 4 - Training Execution:

- Hiển thị progress real-time khi train từng model
- Cache management cho 300k samples
- GPU/CPU detection và auto-switch

Cập nhật Step 5 - Results Analysis:

- So sánh kết quả giữa các model đã train
- Hiển thị Accuracy, Macro-F1, Confusion Matrix
- Export metrics.json, download trained models
- Model performance comparison charts

5) File cần thay đổi

Sửa:

- `models/base/base_model.py`: thêm pipeline API
- `models/register_models.py`: metadata mới (GPU, proba, fusion)
- `training_pipeline.py`: ủy quyền xử lý cho model
- `data_loader.py`: tinh giản, chuyển logic sang TextPreprocessor
- `text_encoders.py`: chuyển logic sang VectorizerProvider
- `app.py`: cập nhật `render_step2_wireframe()` (bỏ preprocessing), `render_step3_wireframe()` (thêm model selector + preprocessing)

Tạo mới:

- `models/utils/{text_preprocessor,vectorizer_provider,feature_fusion,calibration_manager}.py`
- `wizard_ui/components/{model_selector,metrics_panel}.py`

6) Tiêu chí nghiệm thu

Đánh giá trong Streamlit:

- Hiển thị Accuracy, Macro-F1, Confusion Matrix
- Thời gian huấn luyện/suy luận (ước lượng)
- So sánh kết quả giữa các model

UX:

- Tick model → kết quả (có cache) hoặc hiển thị progress
- Spinner/progress rõ ràng, thông báo lỗi thân thiện
- Chạy được CPU, tự động dùng GPU nếu có

7) Lộ trình thực hiện

1. **Tạo utility modules** (TextPreprocessor, VectorizerProvider, FeatureFusion)
2. **Cập nhật BaseModel** với pipeline API

3. **Implement 5 models** theo chuẩn mới (RF, AdaBoost, GBDT, XGB, LGBM)
4. **Cập nhật training_pipeline.py** để dùng model-centric approach
5. **Cập nhật Streamlit UI** trong app.py (Step 3, 4, 5) với model selector và results analysis
6. **Test và tối ưu** performance, UX với 300k samples