Large Language Models in Data Science

Week 4: Text Classification — Intent Routing for AMU Chatbot

Sebastian Mueller
Aix-Marseille Université
2025-2026

Session Overview

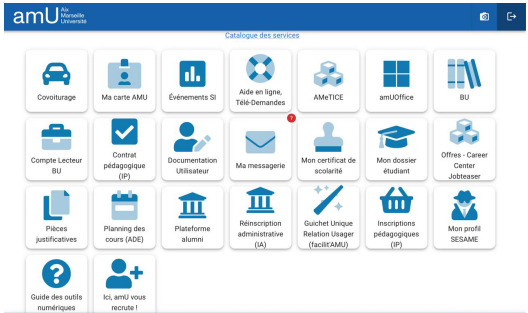
Lecture (15min)

1. Problem framing: intent classification
2. Dataset design and labels
3. Baselines: regex rules
4. Embeddings + Logistic Regression
5. Zero-shot NLI pipeline
6. LLM prompting for classification
7. Comparison and trade-offs

Lab (2.5h)

- Build 4 classifiers (regex, embed+LR, zero-shot, LLM)
- Measure accuracy and latency
- Analyze failure cases and label phrasing
- Extend training data and re-evaluate
- Optional: add a new intent

AMU Chatbot



- AMU is developing a student support chatbot (in French and English).
- The chatbot needs to route student queries to the right AMU service/tool.
- Reliable intent classification is critical for user satisfaction.

Project: AMU Chatbot Brain

- **Goal**: Map student queries to AMU services (*intent routing*).
- **Task**: Given a short query, predict one of 5 intents.
- **Why it matters**: Enables reliable hand-off to the right tool/API.

Intents (lab setup)

- `get_schedule` (Planning des cours / ADE)
- `check_email` (Ma messagerie)
- `register_classes` (Inscriptions pédagogiques / IP)
- `get_student_card` (Ma carte AMU)
- `find_library_info` (BU / Compte Lecteur BU)

Four Approaches at a Glance

- **Regex baseline**: Fastest and transparent; brittle, hard to scale.
- **Embeddings + LogisticRegression**: High accuracy with small data; fast inference; retrain to add intents.
- **Zero-shot classification (MNLI)**: No training; label phrasing matters; moderate latency.
- **LLM prompt**: Most flexible and often strongest; slowest and requires parsing structured output.

Evaluation Plan

- **Metrics**: *Accuracy* (correct label) and *Latency* (ms per query).
- **Protocol**: Run all methods on the same test set; record predictions+timings.
- **Compute note**: Latency varies by device (CPU vs. GPU); compare on same hardware when possible.
- **Reproducibility**: Pin model names/revisions; save prompts and candidates; fix seeds where relevant.

Design Tips

- **Regex**: Prefer precise patterns; avoid overbroad keywords (e.g., plain "amu").
- **Embeddings**: Use multilingual sentence models; validate with cross-validation if data grows.
- **Zero-shot**: Use descriptive, possibly multilingual labels (*emploi du temps* vs. *get_schedule*).
- **LLM prompting**: Specify role, label set, and *strict output format* (JSON / one token).

Key Takeaways & Recommendations

- Start with **embeddings + LR** for a strong, fast baseline.
- Use **zero-shot** for quick prototyping or new intents without data.
- Keep a small **regex layer** for guardrails and trivial routes.
- Add an **LLM fallback** for ambiguous/edge cases; require structured outputs and log decisions.