Τμήμα Πληροφορικής και Τηλεματικής
Χαροκόπειο Πανεπιστήμιο
ΕΠ34 Μηχανική Μάθηση και Εφαρμογές

# 3η Άσκηση: Χρήση μεγάλων προεκπαιδευμένων μοντέλων μετασχηματιστών

Έκδοση 1.1

Διδάσκων: Χρήστος Δίου

## 1 Εισαγωγή

Στην άσκηση αυτή θα χρησιμοποιήσουμε ένα πολύ απλό API για τη χρήση μεγάλων προεκπαιδευμένων μοντέλων μετασχηματιστών (γνωστά και ως μεγάλα γλωσσικά μοντέλα - LLMs) το οποίο βασίζεται στο llama.cpp. Το `llama.cpp` είναι ένα λογισμικό ανοιχτού κώδικα που στοχεύει στην πολύ αποτελεσματική υλοποίηση ανοιχτών LLMs. Είναι γραμμένο σε C++ και σε συνδυασμό με τεχνικές κβαντισμού και αποτελεσματικής προσαρμογής μοντέλων (πχ LoRA) καθιστά την υλοποίηση εφαρμογών βασισμένων σε LLMs προσβάσιμη στο ευρύ κοινό. Η αξία του `llama.cpp` (καθώς και παρόμοιων υλοποιήσεων, όπως το Ollama) ενισχύεται από το συνδυασμό τους με την πλατφόρμα HuggingFace, η οποία προσφέρει ένα πλήρες σύνολο από προεκπαιδευμένα μοντέλα, χωρίς να χρειάζεται να έχουμε μηχανήματα με τεράστιες δυνατότητες (πχ RAM, vRAM, GPU).

### 1.1 Παραδοτέα

Θα πρέπει να παραδώσετε ένα ή περισσότερα αρχεία .ipynb που θα περιέχουν τον κώδικα της υλοποίησής σας, την έξοδο της εκτέλεσής του και τις απαντήσεις στις ερωτήσεις της εργασίας διατυπωμένες σε Markdown στα κελιά ανάμεσα στον κώδικα.

## 2 Εγκατάσταση του llama.cpp

Μπορείτε να εγκαταστήσετε το `llama.cpp` είτε τοπικά στον υπολογιστή σας, ή σε ένα VM του Google Colab. Το αποθετήριο του `llama.cpp` δίνεται στο GitHub. Ωστόσο κάτι τέτοιο δε θα χρειαστεί για την εργασία καθώς θα εγκατασταθεί αυτόματα από το python API που θα χρησιμοποιήσουμε, το `llama-cpp-python`,

```
pip install llama-cpp-python
```

Υπάρχει η δυνατότητα εγκατάστασης έκδοσης του `llama.cpp` η οποία αξιοποιεί την GPU, ωστόσο κάτι τέτοιο δε θα είναι απαραίτητο στα πλαίσια της εργασίας (η CPU αρκεί).

Το `llama.cpp` χρησιμοποιεί μοντέλα που αναπαρίστανται στο format GGUF, το οποίο είναι μία μορφή αρχείου που υποστηρίζει την αποτελεσματική αποθήκευση και χρήση μεγάλων μοντέλων για inference.

## 3 Χρήση μοντέλων ως chatbots

Μελετήστε το high-level API του `llama-cpp-python` module και ειδικά τη συνάρτηση `create_chat_completion`. Αυτή μπορεί να δεχθεί μία σειρά μηνυμάτων που λειτουργεί ως ιστορικό, και παράγει την απάντηση. Παράδειγμα ιστορικού:

```
messages = [
    {"role": "system", "content": "You are a helpful assistant."},
    {"role": "user", "content": "What is the capital of Greece?"},
    {"role": "assistant", "content": "The capital of Greece is Athens."},
    {"role": "user", "content": "Who wrote '20,000 leagues under the sea'?"},
]
```

Χρησιμοποιήστε αυτή την είσοδο και παρατηρήστε την απάντηση για το μοντέλο `MaziyarPanahi/Llama-3.2-3B-Instruct-GGUF` το οποίο βασίζεται στο ανοιχτό μοντέλο Llama 3.2 της Meta. Χρησιμοποιήστε την εκδοχή του μοντέλου που έχει κβαντιστεί στα 4 bit με κωδικοποίηση 'K' και την τεχνική Mixture of Experts (Q4_K_M).

Προετοιμάστε μια σειρά τέτοιων μηνυμάτων ώστε να λάβετε απαντήσεις από το μοντέλο με ποικιλία τρόπων, όπως για παράδειγμα επεξήγηση επιστημονικών εννοιών, συγγραφή κώδικα, χαλαρή συζήτηση και αστεία, φιλοσοφικούς προβληματισμούς, ψυχολογική υποστήριξη, απαντήσεις με ομοιοκαταληξία, απαντήσεις με στίχους ραπ και ότι άλλο φανταστείτε. Πρακτικά τα μηνύματα λειτουργούν ως προτροπές (prompts), μαζί με τα οποία μπορεί να δίνονται και ενδεικτικά παραδείγματα απαντήσεων (χωρίς να είναι απαραίτητο). Σχολιάστε τη συντακτική και εννοιολογική ακρίβεια καθώς και τη συνοχή των απαντήσεων που λαμβάνετε.

## 4 Ερωτήσεις στα Ελληνικά

Κάντε το ίδιο, αλλά αυτή τη φορά δίνοντας τα μηνύματα στα Ελληνικά. Παρατηρήστε τα αποτελέσματα και συγκρίνετε με τα αντίστοιχα αποτελέσματα στα Αγγλικά.

## 5 Εναλλακτικά μοντέλα

Αφού συλλέξετε τις απαντήσεις, χρησιμοποιήστε ένα εναλλακτικό μοντέλο το `bartowski/aya-expanse-8b-GGUF` το οποίο βασίζεται στο μοντέλο AYA της Cohere. Κάντε ότι προηγουμένως, τόσο στα αγγλικά όσο και στα Ελληνικά και συγκρίνετε τα αποτελέσματα (ποιοτικά).

## 6 Ποικιλία απαντήσεων

Καλώντας πολλές φορές την `create_chat_completion` εκτυπώνονται εναλλακτικές απαντήσεις για το ίδιο ερώτημα. Δοκιμάστε το με δύο ενδεικτικά από τα ερωτήματα που ετοιμάσατε στις προηγούμενες ενότητες. Εξηγήστε με ποιους τρόπους μπορεί να επιτευχθεί αυτή η συμπεριφορά. Ποια (ή ποιες) παράμετροι της `create_chat_completion` ελέγχουν την ποικιλία των απαντήσεων που λαμβάνετε από το μοντέλο; Παρουσιάστε μερικά παραδείγματα χρήσης τους.

## 7 Bonus: Χρήση embeddings από LLMs

Η συνάρτηση `create_embedding` του `llama-cpp-python` μας επιτρέπει να λαμβάνουμε ενσωματώσεις κειμένου που παράγει το μοντέλο LLM. Για να δοκιμάσετε αυτή τη λειτουργικότητα κάντε τα εξής:

- Επιλέξτε 5 προτάσεις στα Αγγλικά, με διαφορετικά επίπεδα νοηματικής συνάφειας μεταξύ τους
- Εξάγετε μια λίστα από ενσωματώσεις, μία για κάθε πρόταση
- Υπολογίστε και τυπώστε έναν πίνακα $5 \times 5$, τέτοιον ώστε το στοιχείο $(i, j)$ να αντιστοιχεί στην ομοιότητα συνημιτόνου των ενσωματώσεων της πρότασης $i$ με την πρόταση $j$

Σχολιάστε τα αποτελέσματα και την αντιστοιχία της ομοιότητας των ενσωματώσεων με τη σημασιολογική ομοιότητα.