



ΕΛΛΗΝΙΚΟ ΑΝΟΙΚΤΟ ΠΑΝΕΠΙΣΤΗΜΙΟ

Μεταπτυχιακή Εξειδίκευση
στα Πληροφοριακά Συστήματα

Διπλωματική Εργασία

**Μορφές Εκπαίδευσης Μεγάλων Γλωσσικών Μοντέλων:
Συγκριτική Αξιολόγηση Προσεγγίσεων και Επιδόσεων**

Γεώργιος Κοσμόπουλος

Επιβλέπων Καθηγητής:
Δρ. Δημήτριος Καραπιέρης

ΙΟΥΝΙΟΣ 2026

Η παρούσα εργασία αποτελεί πνευματική ιδιοκτησία του φοιτητή («συγγραφέας/δημιουργός») που την εκπόνησε. Στο πλαίσιο της πολιτικής ανοικτής πρόσβασης ο συγγραφέας/δημιουργός εκχωρεί στο ΕΑΠ, μη αποκλειστική άδεια χρήσης του δικαιώματος αναπαραγωγής, προσαρμογής, δημόσιου δανεισμού, παρουσίασης στο κοινό και ψηφιακής διάχυσής τους διεθνώς, σε ηλεκτρονική μορφή και σε οποιοδήποτε μέσο, για διδακτικούς και ερευνητικούς σκοπούς, άνευ ανταλλάγματος και για όλο το χρόνο διάρκειας των δικαιωμάτων πνευματικής ιδιοκτησίας. Η ανοικτή πρόσβαση στο πλήρες κείμενο για μελέτη και ανάγνωση δεν σημαίνει καθ' οιονδήποτε τρόπο παραχώρηση δικαιωμάτων διανοητικής ιδιοκτησίας του συγγραφέα/δημιουργού ούτε επιτρέπει την αναπαραγωγή, αναδημοσίευση, αντιγραφή, αποθήκευση, πώληση, εμπορική χρήση, μετάδοση, διανομή, έκδοση, εκτέλεση, «μεταφόρτωση» (downloading), «ανάρτηση» (uploading), μετάφραση, τροποποίηση με οποιονδήποτε τρόπο, τμηματικά ή περιληπτικά της εργασίας, χωρίς τη ρητή προηγούμενη έγγραφη συναίνεση του συγγραφέα/δημιουργού. Ο συγγραφέας/δημιουργός διατηρεί το σύνολο των ηθικών και περιουσιακών του δικαιωμάτων.



Μορφές Εκπαίδευσης Μεγάλων Γλωσσικών Μοντέλων: Συγκριτική
Αξιολόγηση Προσεγγίσεων και Επιδόσεων

Γεώργιος Κοσμόπουλος

Επιτροπή Επίβλεψης Διπλωματικής Εργασίας

Επιβλέπων Καθηγητής:

Δημήτριος Καραπιτέρης

Συν-Επιβλέπων Καθηγητής:

Γεώργιος Παπαμιχαήλ

Περίληψη

Η εξέλιξη των Μεγάλων Γλωσσικών Μοντέλων (Large Language Models – LLMs) έχει επηρεάσει ουσιαστικά την Επεξεργασία Φυσικής Γλώσσας, επειδή έφερε στο προσκήνιο μοντέλα που μπορούν να χειριστούν κείμενο με σαφώς μεγαλύτερη ευελιξία από παλαιότερες προσεγγίσεις.

Η παρούσα διπλωματική εξετάζει τρόπους εκπαίδευσης και προσαρμογής Μεγάλων Γλωσσικών Μοντέλων, δίνοντας ιδιαίτερη έμφαση στις τεχνικές αποδοτικής προσαρμογής παραμέτρων (Parameter-Efficient Fine-Tuning – PEFT) και, ειδικότερα, στις LoRA και QLoRA.

Στο πειραματικό μέρος χρησιμοποιήθηκαν ελληνικά δεδομένα και εκτελέσεις τόσο σε τοπικό περιβάλλον όσο και στο Google Colab. Η αξιολόγηση δεν βασίστηκε σε μία μόνο μέτρηση· συνδυάστηκαν το loss, η perplexity, η token-level accuracy, ο χρόνος εκπαίδευσης και η peak VRAM. Έτσι, η ανάλυση δεν περιορίζεται στο αν το μοντέλο προσαρμόζεται στο evaluation set, αλλά εξετάζει και το υπολογιστικό κόστος αυτής της προσαρμογής. Για πληρέστερη εικόνα, προστέθηκαν δοκιμές γενίκευσης σε δεύτερο ελληνικό corpus και ένα μικρό σενάριο DAPT με υλικό από την ελληνική Wikipedia.

Στα περισσότερα πειράματα η QLoRA ήταν σαφώς πιο οικονομική σε VRAM από τη LoRA FP16. Αυτό όμως δεν σημαίνει ότι αποτελεί πάντα την καλύτερη επιλογή. Στο TinyLlama οι διαφορές στην perplexity ήταν μικρές, ενώ στο Gemma-2-2B, κυρίως στην τοπική εκτέλεση, έγινε πιο εμφανές το κόστος της ποσοτικοποίησης. Τα πειράματα γενίκευσης έδωσαν πιο ισορροπημένη εικόνα και δείχνουν ότι η επιλογή μεθόδου εξαρτάται από το τι προέχει κάθε φορά: η μνήμη, ο χρόνος ή η ποσοτική επίδοση. Το DAPT σκέλος έδωσε μια πρώτη ένδειξη ότι μια ενδιάμεση έκθεση σε ελληνικό corpus μπορεί να βελτιώσει προσωρινά τη συμπεριφορά εκτός πεδίου, πριν το επόμενο fine-tuning οδηγήσει ξανά το μοντέλο προς το ειδικότερο dataset.

Λέξεις-Κλειδιά

Μεγάλα Γλωσσικά Μοντέλα, LoRA, QLoRA, PEFT, DAPT, Γενίκευση.

Abstract

This thesis studies parameter-efficient fine-tuning (PEFT) for open-source Large Language Models, with emphasis on LoRA and QLoRA under different compute constraints. The experiments were carried out on a local GPU and on Google Colab, using Greek datasets that include a small controlled Greek text corpus (dataset_v1) and a Wikipedia-derived corpus (dataset_wiki_50k). The evaluation combines cross-entropy loss, perplexity and token-level accuracy with more practical measurements, such as training time and peak GPU memory. The results indicate that QLoRA consistently reduces memory usage compared with LoRA FP16, although its effect on quality and runtime depends on the model, sequence length and hardware configuration. In the reverse-generalization setting (wiki→v1), the two methods produced almost identical generalization perplexity.

A further set of A100 experiments was included for models and configurations that were difficult or impossible to complete locally. These runs are not meant to form a full ranking of models. They are used as a practical check of how memory use, runtime and evaluation metrics change when both the model family and the available hardware change. Taken together, the experiments provide a reproducible view of LoRA and QLoRA behaviour in realistic Greek-language fine-tuning settings.

Keywords

Large Language Models, LoRA, QLoRA, PEFT, DAPT, Generalization.

Περιεχόμενα

Περίληψη	iv
Abstract	v
Κατάλογος Εικόνων / Σχημάτων	ix
Κατάλογος Πινάκων	x
Συντομογραφίες και Ακρωνύμια	xi
1. Εισαγωγή	1
1.1 Πλαίσιο και κίνητρο.....	1
1.2 Στόχος και ερευνητικά ερωτήματα	1
1.3 Δομή εργασίας	2
2. Θεωρητικό υπόβαθρο και βιβλιογραφική ανασκόπηση	3
2.1 Μεγάλα Γλωσσικά Μοντέλα και στόχος next-token prediction.....	3
2.2 Αρχιτεκτονική Transformer	3
2.2.1 Υπολογιστική πολυπλοκότητα και κλιμάκωση	4
2.3 Μορφές εκπαίδευσης LLMs	4
2.3.1 Pretraining και continual pretraining	4
2.3.2 Fine-tuning, instruction tuning και RLHF	4
2.4 Κόστος πλήρους fine-tuning.....	5
2.5 Parameter-Efficient Fine-Tuning (PEFT)	5
2.6 LoRA (Low-Rank Adaptation)	5
2.7 Ποσοτικοποίηση και QLoRA	5
2.7.1 NF4, double quantization και paged optimizers	6
2.8 Μετρικές αξιολόγησης.....	6
2.9 Μοντέλα αναφοράς στην εργασία	6
2.10 Πειραματική υλοποίηση και ρυθμίσεις εκπαίδευσης	7
2.11 Attention kernels, SDPA και συμβατότητα Gemma-2	7
2.12 Θέση των LoRA/QLoRA στο ευρύτερο πλαίσιο PEFT	7
3. Σχεδιασμός πειραμάτων.....	9
3.1 Υλικό και υπολογιστικά περιβάλλοντα	9
3.2 Λογισμικό και εργαλεία.....	9
3.3 Μοντέλα.....	9
3.4 Σύνολα δεδομένων και προεπεξεργασία.....	10
3.5 Πειραματικός σχεδιασμός και υπερπαράμετροι	10
3.5.1 Άξονες πειραματικής αξιολόγησης.....	12
3.5.2 Στόχοι της πειραματικής αξιολόγησης	12

3.6 Μετρικές και αρχεία αποτελεσμάτων	13
4. Αποτελέσματα (RTX 3070) και πλαίσια σύγκρισης με Colab	15
4.1 Σύνοψη πειραμάτων RTX 3070.....	15
4.2 Γραφήματα (RTX 3070)	16
4.3 Καμπύλες σύγκλισης (Loss Curves, r=16)	17
4.4 Παρατηρήσεις από τα αποτελέσματα RTX 3070	19
4.5 Πλαίσιο ενοποίησης για σύγκριση με Colab	19
4.6 Ρυθμίσεις πειραμάτων Colab (GPU L4).....	20
4.7 Πίνακας αποτελεσμάτων Colab (GPU L4).....	20
4.8 Σύγκριση Colab L4 έναντι RTX 3070 (πλαίσιο ερμηνείας).....	21
4.9 Πίνακας αποτελεσμάτων Colab (GPU A100)	22
4.10 Σύγκριση Colab L4 έναντι Colab A100 (αναπαραγωγή πειραμάτων)	22
4.11 Συγκεντρωτική οπτικοποίηση απόδοσης και κόστους	23
4.12 Διερεύνηση ορίων στο Colab (GPU A100).....	27
4.13 Έλεγχος γενίκευσης (dataset_v1 → dataset_wiki_50k)	29
4.14 Αντίστροφος έλεγχος γενίκευσης (dataset_wiki_50k → dataset_v1)	30
4.15 Συνεχιζόμενη προεκπαίδευση τύπου DAPT σε ελληνικό corpus.....	30
4.16 Διερεύνηση κλιμάκωσης σε A100 80GB (VRAM).....	32
5. Συζήτηση.....	35
5.1 LoRA έναντι QLoRA: αντιστάθμιση ποιότητας, μνήμης και χρόνου	35
5.2 Επίδραση του rank r.....	35
5.3 Επίδραση υπολογιστικού περιβάλλοντος	35
5.4 Πλαίσιο επιλογής τεχνικής ανά διαθέσιμους πόρους	36
5.5 Περιορισμοί και απειλές εγκυρότητας.....	37
6. Συμπεράσματα	40
Βιβλιογραφία	41
Ελληνική βιβλιογραφία.....	41
Ξενόγλωσση βιβλιογραφία (APA).....	41
Παράρτημα Α: Κώδικας εκτέλεσης πειραμάτων και ενοποίησης αποτελεσμάτων.....	45
A.1 Master runner (THESIS_MASTER_RUNS.py)	45
A.2 Runner script (run_peft_experiment.LOCAL.colab.py).....	45
A.3 Post-processing (postprocess_all_runs.py)	46
A.4 Συγκεντρωτικός πίνακας (make_master_results.py)	46
Παράρτημα Β: Ενδεικτικές εντολές εκτέλεσης και αναπαραγωγής.....	48
B.1 Εκτέλεση run-set μέσω master runner	48

B.2 Baseline fine-tuning (dataset_v1).....	48
B.3 Generalization evaluation (προαιρετικό)	48
B.4 Reverse generalization (train: dataset_wiki_50k → eval: dataset_v1)	48
B.5 DAPT (continued pretraining-like) και fine-tuning	48
B.6 Ρυθμίσεις σταθερότητας (προαιρετικά)	49
Παράρτημα Γ: Δομή φακέλων αποτελεσμάτων και παραγόμενα αρχεία.....	50
Γ.1 Δομή φακέλων.....	50
Γ.2 Περιγραφή βασικών αρχείων	50

Κατάλογος Εικόνων / Σχημάτων

- Σχήμα 4.1: Perplexity ως συνάρτηση του rank r (dataset_v1, RTX 3070). (σελ. 16)
- Σχήμα 4.2: Χρόνος εκπαίδευσης ως συνάρτηση του rank r (dataset_v1, RTX 3070). (σελ. 16)
- Σχήμα 4.3: Μέγιστη χρήση VRAM για $r=16$ (dataset_v1, RTX 3070). (σελ. 17)
- Σχήμα 4.4: TinyLlama – LoRA FP16 ($r=16$): καμπύλη training loss. (σελ. 17)
- Σχήμα 4.5: TinyLlama – QLoRA ($r=16$): καμπύλη training loss. (σελ. 18)
- Σχήμα 4.6: Gemma-2-2B – LoRA FP16 ($r=16$): καμπύλη training loss. (σελ. 18)
- Σχήμα 4.7: Gemma-2-2B – QLoRA ($r=16$): καμπύλη training loss. (σελ. 19)
- Σχήμα 4.8: Μέγιστη χρήση VRAM ανά μέθοδο και μοντέλο. (σελ. 24)
- Σχήμα 4.9: Χρόνος εκπαίδευσης ανά μέθοδο και μοντέλο. (σελ. 24)
- Σχήμα 4.10: Perplexity ανά μέθοδο και μοντέλο. (σελ. 25)
- Σχήμα 4.11: Σχέση peak VRAM και perplexity. (σελ. 25)
- Σχήμα 4.12: Σχέση χρόνου εκπαίδευσης και perplexity. (σελ. 26)
- Σχήμα 4.13: Generalization perplexity σε πειράματα $v1 \rightarrow \text{wiki}$ και $\text{wiki} \rightarrow v1$. (σελ. 26)
- Σχήμα 4.14: Σύγκριση baseline fine-tuning, DAPT και DAPT $\rightarrow$ fine-tuning ως προς wiki generalization perplexity. (σελ. 27)

Κατάλογος Πινάκων

Πίνακας 3.1: Μοντέλα, βασικά χαρακτηριστικά και αιτιολόγηση επιλογής. (σελ. 10)

Πίνακας 4.1: Συνοπτικά αποτελέσματα RTX 3070 (dataset_v1 και dataset_wiki_50k). (σελ. 15)

Πίνακας 4.2: Συνοπτικά αποτελέσματα Colab L4 (dataset_v1). (σελ. 20)

Πίνακας 4.3: Σύγκριση Gemma-2-2B υπό ίδιες ρυθμίσεις ($r=16$, $\text{max_length}=128$) σε RTX 3070, Colab L4 και Colab A100. (σελ. 21)

Πίνακας 4.4: Συνοπτικά αποτελέσματα Colab A100 (dataset_v1). (σελ. 22)

Πίνακας 4.5: Αποτελέσματα διερεύνησης ορίων στο A100 (Gemma-2-2B, dataset_v1). (σελ. 28)

Πίνακας 4.6: Πειράματα γενίκευσης (L4) – in-domain και out-of-domain metrics (wiki_50k). (σελ. 29)

Πίνακας 4.7: Αντίστροφος έλεγχος γενίκευσης (dataset_wiki_50k $\rightarrow$ dataset_v1) – αποτελέσματα (L4). (σελ. 30)

Πίνακας 4.8: Αποτελέσματα DAPT και Stage-2 (DAPT $\rightarrow$ fine-tuning) σε ελληνικό corpus. (σελ. 31)

Πίνακας 4.9: Πείραμα κλιμάκωσης σε A100 80GB – VRAM scaling (Len=1024, Batch=64). (σελ. 32)

Πίνακας 4.10: Συμπληρωματικά αποτελέσματα πρόσθετων μοντέλων σε Colab A100. (σελ. 33)

Συντομογραφίες και Ακρωνύμια

- LLM: Large Language Model (Μεγάλο Γλωσσικό Μοντέλο)
- PEFT: Parameter-Efficient Fine-Tuning
- LoRA: Low-Rank Adaptation
- QLoRA: Quantized LoRA (4-bit βάση + LoRA adapters)
- VRAM: Μνήμη κάρτας γραφικών
- FP16: Half precision (float16)
- NF4: NormalFloat4 (μορφή 4-bit quantization)
- SDPA: Scaled Dot-Product Attention (kernel στο PyTorch)
- r : rank της LoRA (υπερπαράμετρος χωρητικότητας)
- DAPT: Domain-Adaptive PreTraining (συνεχιζόμενη προεκπαίδευση σε domain δεδομένα)

1. Εισαγωγή

1.1 Πλαίσιο και κίνητρο

Η ανάπτυξη Μεγάλων Γλωσσικών Μοντέλων (Large Language Models – LLMs) έχει επιφέρει ριζικές αλλαγές στην Επεξεργασία Φυσικής Γλώσσας (NLP), με τα μοντέλα τύπου Transformer να αποτελούν πλέον το βασικό αρχιτεκτονικό υπόβαθρο (Vaswani et al., 2017). Σε πρακτικό επίπεδο, όμως, η αξιοποίησή τους προϋποθέτει επιλογές εκπαίδευσης που έχουν πραγματικό κόστος: κάθε στρατηγική απαιτεί διαφορετική μνήμη, χρόνο και υπολογιστικό προϋπολογισμό, ενώ δεν οδηγεί πάντα στην ίδια ποιότητα προσαρμογής. Το ζήτημα γίνεται ακόμη πιο κρίσιμο όταν ο στόχος είναι η χρήση LLMs σε περιβάλλοντα περιορισμένων πόρων ή σε γλώσσες με μικρότερο οικοσύστημα δεδομένων, όπως η ελληνική.

Από αυτή την αφετηρία, η εργασία δεν μένει μόνο σε μια θεωρητική χαρτογράφηση των μορφών εκπαίδευσης των LLMs. Εξετάζει ταυτόχρονα ένα πρακτικό ερώτημα: ποιες επιλογές προσαρμογής παραμένουν ρεαλιστικές όταν οι διαθέσιμοι πόροι είναι περιορισμένοι και το πείραμα πρέπει να στηθεί σε πραγματικές συνθήκες, χωρίς να θυσιάζεται υπερβολικά η ποιότητα του αποτελέσματος.

1.2 Στόχος και ερευνητικά ερωτήματα

Στόχος της εργασίας είναι, αφενός, να παρουσιάσει με οργανωμένο τρόπο τις βασικές μορφές εκπαίδευσης LLMs (pretraining, continual pretraining, fine-tuning, instruction tuning, RLHF) και, αφετέρου, να εξετάσει πειραματικά πώς συμπεριφέρονται στην πράξη δύο διαδεδομένες PEFT τεχνικές, η LoRA και η QLoRA, όταν εφαρμόζονται σε ελληνικά δεδομένα και σε ετερογενή υπολογιστικά περιβάλλοντα.

Πιο συγκεκριμένα, η εργασία εξετάζει τα εξής ερευνητικά ερωτήματα: (α) πώς συγκρίνονται η LoRA και η QLoRA ως προς την ποιότητα προσαρμογής (loss, perplexity, token-level accuracy), το υπολογιστικό κόστος (χρόνος εκπαίδευσης) και τις απαιτήσεις μνήμης (VRAM), (β) πώς επηρεάζει το υπολογιστικό περιβάλλον, δηλαδή η τοπική RTX 3070 σε σχέση με Colab L4/A100, την ερμηνεία αυτών των μετρήσεων, και (γ) κατά πόσο προκύπτει μια πρακτική γραμμή επιλογής μεθόδου ανάλογα με τους περιορισμούς πόρων και τις ανάγκες γενίκευσης.

1.3 Δομή εργασίας

Στο Κεφάλαιο 2 παρουσιάζονται το θεωρητικό υπόβαθρο και η βιβλιογραφική ανασκόπηση για την αρχιτεκτονική Transformer, τις κύριες μορφές εκπαίδευσης LLMs και τις τεχνικές PEF. Το Κεφάλαιο 3 περιγράφει τη μεθοδολογία, τα μοντέλα, τα σύνολα δεδομένων, τους άξονες αξιολόγησης και τις μετρικές. Στο Κεφάλαιο 4 αναλύονται τα αποτελέσματα από τις τοπικές εκτελέσεις (RTX 3070) και από τα περιβάλλοντα Colab. Στα Κεφάλαια 5 και 6 ακολουθούν η συζήτηση των ευρημάτων και τα συμπεράσματα. Η συγγραφή και η μορφοποίηση της εργασίας ακολουθούν τις προδιαγραφές του ΕΑΠ (Ελληνικό Ανοικτό Πανεπιστήμιο, 2018) και τον ειδικό κανονισμό ΜΔΕ ΠΛΣ (Ελληνικό Ανοικτό Πανεπιστήμιο, 2020).

2. Θεωρητικό υπόβαθρο και βιβλιογραφική ανασκόπηση

Η βιβλιογραφική ανασκόπηση του κεφαλαίου οργανώνεται γύρω από τέσσερις άξονες που συνδέονται άμεσα με την εργασία: το αρχιτεκτονικό και μαθησιακό υπόβαθρο των LLMs, τις βασικές μορφές εκπαίδευσής τους, τις τεχνικές PEFT με έμφαση στις LoRA και QLoRA, και τη θέση των επιλεγμένων μοντέλων μέσα στο ευρύτερο ερευνητικό τοπίο. Δεν επιχειρείται πλήρης κάλυψη όλης της βιβλιογραφίας των LLMs. Η ανασκόπηση εστιάζει στα σημεία που είναι απαραίτητα για να στηριχθεί η πειραματική μελέτη που ακολουθεί.

2.1 Μεγάλα Γλωσσικά Μοντέλα και στόχος next-token prediction

Τα LLMs εκπαιδεύονται συνήθως ως γλωσσικά μοντέλα που μεγιστοποιούν την πιθανότητα μιας ακολουθίας tokens, προβλέποντας το επόμενο token (next-token prediction). Μέσω της εκμάθησης αυτής της πιθανοτικής κατανομής, αναπτύσσονται αναδυόμενες ικανότητες που μεταφέρονται σε πολλές εργασίες NLP (Raffel et al., 2020). Η εξέλιξη των μεγάλων decoder-only μοντέλων, από τα GPT/GPT-2 έως τα GPT-3, ανέδειξε τη σημασία της κλιμάκωσης παραμέτρων, δεδομένων και υπολογισμού, καθώς και την ικανότητα few-shot προσαρμογής χωρίς πλήρες fine-tuning (Radford et al., 2019· Brown et al., 2020· Kaplan et al., 2020).

2.2 Αρχιτεκτονική Transformer

Ο Transformer εισήχθη ως αρχιτεκτονική που βασίζεται αποκλειστικά σε μηχανισμούς προσοχής, επιτρέποντας πλήρη παραλληλοποίηση έναντι των RNN/LSTM (Vaswani et al., 2017). Για LLMs παραγωγής κειμένου χρησιμοποιείται κατά κύριο λόγο η decoder-only παραλλαγή, όπου η προσοχή είναι causally masked ώστε κάθε θέση να «βλέπει» μόνο το παρελθόν.

Κεντρικός μηχανισμός είναι το scaled dot-product attention. Για δεδομένα Q, K, V, ορίζεται ως: $\text{Attention}(Q, K, V) = \text{softmax}((QK^T)/\sqrt{d_k}) V$ (Vaswani et al., 2017). Στο multi-head attention, το attention υπολογίζεται σε πολλαπλές κεφαλές, επιτρέποντας μάθηση διαφορετικών τύπων συσχέτισης. Επιπλέον, χρησιμοποιούνται positional embeddings/encodings, residual connections και layer normalization για σταθερότητα.

Σε νεότερες παραλλαγές των decoder-only LLMs χρησιμοποιούνται επιπλέον τεχνικές που βελτιώνουν την αποδοτικότητα και τη σταθερότητα, όπως rotary positional embeddings (RoPE) για την κωδικοποίηση θέσης και παραλλαγές gated feed-forward layers, όπως GLU/SwiGLU, στο εσωτερικό των Transformer blocks (Shazeer, 2020· Su et al., 2021).

2.2.1 Υπολογιστική πολυπλοκότητα και κλιμάκωση

Το self-attention έχει πολυπλοκότητα $O(n^2)$ ως προς το μήκος ακολουθίας n , γεγονός που επηρεάζει άμεσα τη μνήμη και τον χρόνο κατά την εκπαίδευση. Για τον λόγο αυτό, το `max_length` αποτελεί κρίσιμη υπερπαράμετρο σε περιβάλλοντα περιορισμένης VRAM.

2.3 Μορφές εκπαίδευσης LLMs

Στη βιβλιογραφία διακρίνονται οι κύριες μορφές εκπαίδευσης: (α) pretraining από το μηδέν, (β) continual pretraining (domain adaptation), (γ) fine-tuning για συγκεκριμένα καθήκοντα, (δ) instruction tuning, και (ε) RLHF για ευθυγράμμιση με ανθρώπινες προτιμήσεις (Raffel et al., 2020· Ouyang et al., 2022).

Η εξέλιξη των μεγάλων προεκπαιδευμένων μοντέλων περιλαμβάνει τόσο encoder-only προσεγγίσεις, όπως το BERT, όσο και μεγάλα ανοικτά decoder-only ή multilingual μοντέλα, όπως τα OPT και BLOOM. Οι διαφορετικές αυτές γραμμές έρευνας δείχνουν ότι η επιλογή αρχιτεκτονικής και στρατηγικής εκπαίδευσης εξαρτάται από τον στόχο χρήσης, το διαθέσιμο corpus και τον υπολογιστικό προϋπολογισμό (Devlin et al., 2019· Zhang et al., 2022· Scao et al., 2022).

2.3.1 Pretraining και continual pretraining

Το pretraining πραγματοποιείται σε μεγάλα corpora και αποσκοπεί στην εκμάθηση γενικών γλωσσικών προτύπων. Η συνεχιζόμενη προεκπαίδευση (continual pretraining) αποτελεί επέκταση του pretraining σε νεότερα ή εξειδικευμένα δεδομένα, με στόχο τη μεταφορά σε συγκεκριμένο domain χωρίς πλήρες retraining (Gururangan et al., 2020).

2.3.2 Fine-tuning, instruction tuning και RLHF

Το fine-tuning προσαρμόζει ένα pretrained μοντέλο σε συγκεκριμένο task/dataset. Το instruction tuning χρησιμοποιεί δεδομένα τύπου «εντολή → απάντηση» ώστε το μοντέλο να ακολουθεί οδηγίες, ενώ το RLHF χρησιμοποιεί ανθρώπινη ανατροφοδότηση/προτιμήσεις για ευθυγράμμιση των εξόδων (Ouyang et al., 2022).

Στην πράξη, το instruction tuning υλοποιείται ως supervised fine-tuning πάνω σε σύνολα εντολών/απαντήσεων (instruction–response pairs). Η βιβλιογραφία έχει δείξει ότι η εκπαίδευση σε πολλαπλά καθήκοντα και η διατύπωση προβλημάτων ως οδηγίες μπορούν να ενισχύσουν τη zero-shot και cross-task γενίκευση (Sanh et al., 2022· Wei et al., 2022· Chung et al., 2022). Ενδεικτικά παραδείγματα αποτελούν τα FLAN/FLAN-T5 (Chung et al., 2022), Stanford Alpaca (Taori et al., 2023), Databricks Dolly (Conover et al., 2023) και Zephyr

(Tunstall et al., 2023), τα οποία χρησιμοποιούνται είτε για άμεσο instruction fine-tuning είτε ως στάδιο πριν/αντί του RLHF.

2.4 Κόστος πλήρους fine-tuning

Στο πλήρες fine-tuning ενημερώνονται όλα τα βάρη του μοντέλου. Πέρα από τα βάρη, απαιτείται αποθήκευση gradients και optimizer states, με αποτέλεσμα το memory footprint να αυξάνεται σημαντικά σε σχέση με το inference. Αυτό καθιστά δύσκολη την προσαρμογή μεγάλων μοντέλων σε GPUs περιορισμένης μνήμης.

2.5 Parameter-Efficient Fine-Tuning (PEFT)

Οι τεχνικές PEFT «παγώνουν» το βασικό μοντέλο και εκπαιδεύουν μικρές πρόσθετες δομές (adapters, prompts, low-rank ενημερώσεις), μειώνοντας δραστικά τις εκπαιδευσιμες παραμέτρους και την απαιτούμενη VRAM (Hu et al., 2021). Στο ίδιο πλαίσιο εντάσσονται και τεχνικές όπως το prefix-tuning και το prompt tuning/P-Tuning v2, οι οποίες βελτιστοποιούν μικρό αριθμό πρόσθετων παραμέτρων αντί να ενημερώνουν ολόκληρο το μοντέλο (Li & Liang, 2021· Liu et al., 2022).

2.6 LoRA (Low-Rank Adaptation)

Η LoRA εισάγει low-rank ενημέρωση ΔW σε επιλεγμένα βάρη W μέσω παραγοντοποίησης $\Delta W = A \cdot B$, με $\text{rank } r \ll \min(d, k)$. Εκπαιδεύονται μόνο οι παράμετροι των A και B , διατηρώντας παγωμένα τα αρχικά βάρη (Hu et al., 2021). Η υπερπαράμετρος r ελέγχει τη χωρητικότητα προσαρμογής και το κόστος. Στην πράξη, η LoRA εφαρμόζεται συχνά σε attention projections όπως q_proj και v_proj .

2.7 Ποσοτικοποίηση και QLoRA

Η ποσοτικοποίηση μειώνει την ακρίβεια αναπαράστασης των βαρών (π.χ. από FP16 σε 4-bit), μειώνοντας τη μνήμη και επιτρέποντας fine-tuning μεγαλύτερων μοντέλων σε περιορισμένη VRAM. Η QLoRA συνδυάζει 4-bit quantization (π.χ. NF4) με LoRA adapters, επιτυγχάνοντας σημαντική μείωση μνήμης με μικρή ή και μηδενική απώλεια απόδοσης σε αρκετές ρυθμίσεις (Dettmers et al., 2023).

2.7.1 NF4, double quantization και paged optimizers

Στην QLoRA χρησιμοποιούνται τεχνικές όπως το NormalFloat4 (NF4), το double quantization και οι paged optimizers. Ο ρόλος τους είναι να περιορίζουν αιχμές μνήμης και να αξιοποιούν πιο αποδοτικά τη διαθέσιμη VRAM (Dettmers et al., 2023).

2.8 Μετρικές αξιολόγησης

Για την αξιολόγηση μοντέλων γλώσσας, βασική μετρική είναι το perplexity, που ορίζεται ως $\exp(\text{eval_loss})$. Χαμηλότερο perplexity υποδηλώνει καλύτερη πιθανοτική πρόβλεψη επόμενων tokens. Συμπληρωματικά, χρησιμοποιείται token-level accuracy ως ενδεικτική μέτρηση ακρίβειας. Για την αποδοτικότητα καταγράφονται ο χρόνος εκπαίδευσης και η μέγιστη χρήση μνήμης GPU (peak_vram_mib).

2.9 Μοντέλα αναφοράς στην εργασία

Η εργασία χρησιμοποιεί δύο μοντέλα: το TinyLlama (1.1B), ένα μικρό LLM συμβατό με την οικογένεια Llama (Zhang et al., 2024), και το Gemma-2-2B, ένα πρακτικό ανοικτό μοντέλο μεγέθους 2B (Gemma Team, 2024). Η επιλογή τους δεν έγινε για να αναδειχθεί κάποιο «καλύτερο» foundation model, αλλά για να εξεταστεί αν τα συμπεράσματα για τις LoRA και QLoRA παραμένουν σταθερά όταν αλλάζει η κλίμακα του μοντέλου και το διαθέσιμο περιθώριο μνήμης. Το TinyLlama λειτουργεί ως ελαφρύ και ρεαλιστικό σημείο αναφοράς για ευρέως διαθέσιμες GPUs καταναλωτικής κατηγορίας, ενώ το Gemma-2-2B επιτρέπει να δοκιμαστεί η ίδια μεθοδολογία σε απαιτητικότερο μοντέλο.

Για την ελληνική γλώσσα υπάρχει πλέον μικρός αλλά αυξανόμενος αριθμός σχετικών μοντέλων και πόρων. Ενδεικτικά αναφέρονται το GREEK-BERT (Koutsikakis et al., 2020), το Meltemi 7B (Voukoutis et al., 2024), το Krikri (Roussis et al., 2025) και sequence-to-sequence προσεγγίσεις όπως το GreekBART (Evdaïmon et al., 2024). Ως γλωσσικός πόρος αναφέρεται επίσης το Hellenic National Corpus (Gavrilidou, 2002). Στην παρούσα εργασία τα Meltemi 7B και Krikri δεν χρησιμοποιούνται για να εξαχθεί γενική κατάταξη ελληνικών LLMs. Εντάσσονται πιο περιορισμένα ως πρόσθετες δοκιμές QLoRA μέσα στο συγκεκριμένο πειραματικό πλαίσιο.

Πέρα από τα δύο κύρια μοντέλα, έγιναν συμπληρωματικές εκτελέσεις με Qwen2.5-1.5B-Instruct (Qwen Team, 2024), Phi-3.5 Mini (Microsoft, 2024), Meltemi 7B και Krikri 8B. Τα Qwen και Phi επιλέχθηκαν ως πρόσθετα instruction-tuned μοντέλα μικρής και μεσαίας

κλίμακας, επειδή σε αυτά ήταν εφικτή η πιο άμεση σύγκριση LoRA FP16 και QLoRA. Τα Meltani 7B και Krikri 8B είχαν διαφορετικό ρόλο: χρησιμοποιήθηκαν κυρίως ως δοκιμές εφικτότητας για μεγαλύτερα ελληνικά ή ελληνόφωνα μοντέλα, ώστε να φανεί στην πράξη μέχρι ποιο σημείο βοηθά η QLoRA όταν αυξάνονται το μέγεθος του μοντέλου και οι απαιτήσεις μνήμης.

2.10 Πειραματική υλοποίηση και ρυθμίσεις εκπαίδευσης

Η πειραματική υλοποίηση βασίστηκε στο οικοσύστημα Hugging Face (Transformers + PEFT) και στο Trainer API. Για λόγους επαναληψιμότητας, κάθε εκτέλεση αποθήκευε τα ίδια βασικά αρχεία αποτελεσμάτων (metrics.json, results_table.csv, log_history.csv, loss_curve.png), μαζί με μετρήσεις για τη μέγιστη χρήση VRAM και τον χρόνο εκπαίδευσης. Χρησιμοποιήθηκαν επίσης επιλογές σταθερότητας, όπως gradient checkpointing και απενεργοποίηση του use_cache κατά την εκπαίδευση. Ως βασικό πλαίσιο βελτιστοποίησης χρησιμοποιήθηκαν παραλλαγές AdamW, δηλαδή decoupled weight decay optimization, που αποτελεί καθιερωμένη επιλογή για σύγχρονη εκπαίδευση νευρωνικών μοντέλων (Loshchilov & Hutter, 2019). Στις εκτελέσεις QLoRA χρησιμοποιήθηκε paged optimizer, με στόχο να περιοριστούν οι αιχμές μνήμης σε περιβάλλοντα με περιορισμένη VRAM.

2.11 Attention kernels, SDPA και συμβατότητα Gemma-2

Στον PyTorch/Transformers, το self-attention μπορεί να εκτελεστεί μέσω διαφορετικών kernels (π.χ. SDPA/Flash-style kernels ή eager fallback). Σε ορισμένους συνδυασμούς ποσοτικοποιημένων βαρών και adapters (ιδιαίτερα σε Gemma-2 με 4-bit quantization), μπορεί να εμφανιστούν ασυμβατότητες dtype στη διαδρομή SDPA. Για να διασφαλιστεί σταθερότητα και αναπαραγωγιμότητα, στις σχετικές εκτελέσεις επιλέχθηκε eager attention όπου απαιτήθηκε. Η επιλογή του kernel μπορεί να επηρεάσει τον χρόνο εκτέλεσης· για αυτό στις συγκρίσεις γίνεται σαφής αναφορά όταν χρησιμοποιούνται ρυθμίσεις συμβατότητας.

2.12 Θέση των LoRA/QLoRA στο ευρύτερο πλαίσιο PEFT

Οι τεχνικές PEFT περιλαμβάνουν, πέρα από LoRA/QLoRA, προσεγγίσεις όπως adapters, prompt/prefix tuning και IA³. Οι μέθοδοι prefix/prompt tuning έδειξαν ότι η προσαρμογή μπορεί να επιτευχθεί με πολύ μικρότερο αριθμό εκπαιδευσιμων παραμέτρων, ιδίως όταν το βασικό μοντέλο παραμένει σταθερό (Li & Liang, 2021· Liu et al., 2022). Στην παρούσα εργασία επιλέχθηκαν LoRA και QLoRA επειδή αποτελούν από τις πιο διαδεδομένες και

πρακτικές τεχνικές PEFT για LLMs, με καλή υποστήριξη εργαλείων και δυνατότητα εφαρμογής σε διαφορετικά μεγέθη μοντέλων. Η QLoRA επεκτείνει την LoRA με 4-bit quantized βάση, επιτρέποντας fine-tuning όταν η VRAM είναι ο κύριος περιοριστικός πόρος.

3. Σχεδιασμός πειραμάτων

3.1 Υλικό και υπολογιστικά περιβάλλοντα

Οι εκτελέσεις έγιναν σε δύο βασικά περιβάλλοντα: (α) τοπικό σύστημα με NVIDIA GeForce RTX 3070, 8GB VRAM, 32GB RAM και AMD Ryzen 5 7600X, και (β) Google Colab Pro με NVIDIA L4 24GB. Για τις συμπληρωματικές εκτελέσεις χρησιμοποιήθηκαν επίσης GPUs NVIDIA A100 μέσω Google Colab, σε διαμορφώσεις 40GB και 80GB ανάλογα με το πείραμα. Οι βασικές εκτελέσεις αναπαραγωγής και διερεύνησης κλιμάκωσης πραγματοποιήθηκαν σε A100 80GB, ενώ οι πρόσθετες δοκιμές με Qwen2.5, Phi-3.5, Meltimi και Krikri πραγματοποιήθηκαν σε A100 40GB.

3.2 Λογισμικό και εργαλεία

Για την υλοποίηση χρησιμοποιήθηκαν οι βιβλιοθήκες PyTorch, Hugging Face Transformers/Datasets, PEFT και bitsandbytes για ποσοτικοποίηση 4-bit (Wolf et al., 2020).

Για την αυτοματοποίηση στο Colab χρησιμοποιήθηκαν βασικά σενάρια εκτέλεσης για διερεύνηση παραμέτρων (2 μέθοδοι $\times$ 3 τιμές rank) που καλούν κοινό πρόγραμμα εκπαίδευσης/αξιολόγησης. Το πρόγραμμα οργανώνει τα αποτελέσματα ανά run_id σε τυποποιημένη δομή φακέλων και παράγει αρχεία αποτελεσμάτων (metrics.json, results_table.csv κ.λπ.).

3.3 Μοντέλα

Τα μοντέλα που αξιολογούνται είναι: (α) TinyLlama/TinyLlama-1.1B-Chat-v1.0 και (β) google/gemma-2-2b-it. Τα βασικά χαρακτηριστικά τους και η αιτιολόγηση της επιλογής τους παρουσιάζονται στον Πίνακα 3.1. Η επιλογή τους υπηρετεί τη διερεύνηση της εφικτότητας PEFT σε περιορισμένους πόρους και τον έλεγχο της σταθερότητας των συμπερασμάτων μεταξύ μικρότερου και μεγαλύτερου μοντέλου.

Μοντέλο	Μέγεθος	Τύπος	Κύρια χαρακτηριστικά	Λόγος επιλογής
TinyLlama/ TinyLlama- 1.1B-Chat-v1.0	~1.1B	Instruction- tuned διάλογος	Μικρό ανοικτό μοντέλο, συμβατό με την οικογένεια Llama, κατάλληλο για πειράματα σε περιορισμένους πόρους.	Επιτρέπει γρήγορους επαναληπτικούς ελέγχους και δοκιμές εφικτότητας σε RTX 3070 8GB.
google/ gemma-2-2b-it	~2B	Instruction- tuned	Μεγαλύτερο ανοικτό μοντέλο με αυξημένες απαιτήσεις μνήμης και μεγαλύτερο πρακτικό ενδιαφέρον.	Λειτουργεί ως απαιτητικότερο σημείο αναφοράς, ώστε να ελεγχθεί αν τα συμπεράσματα διατηρούνται σε μεγαλύτερη κλίμακα.

Πίνακας 3.1: Μοντέλα, βασικά χαρακτηριστικά και αιτιολόγηση επιλογής.

3.4 Σύνολα δεδομένων και προεπεξεργασία

Χρησιμοποιήθηκαν δύο ελληνικά σύνολα δεδομένων: το dataset_v1 (μικρό σύνολο για γρήγορα βασικά πειράματα) και το dataset_wiki_50k (μεγαλύτερο σύνολο περίπου 50 χιλιάδων δειγμάτων από τη Wikipedia για έλεγχο κλιμάκωσης). Τα δεδομένα αποθηκεύονται ως απλό κείμενο (train_el.txt) και φορτώνονται με τη βιβλιοθήκη Hugging Face Datasets (Lhoest et al., 2021). Ο διαχωρισμός γίνεται σε train/test μέσω της παραμέτρου eval_test_size.

Επιπλέον, δημιουργήθηκε το dataset_pretrain_el (Greek Wikipedia-derived corpus) ως σύνολο δεδομένων τύπου προεκπαίδευσης για πειράματα συνεχιζόμενης προεκπαίδευσης (DAPT). Το corpus κατασκευάστηκε από το wikimedia/wikipedia (20231101.el) (Wikimedia Foundation, 2023) με τεμαχισμό σε τμήματα κειμένου και βασικό φιλτράρισμα μήκους (ελάχιστο μήκος 120 χαρακτήρες), ώστε να είναι κατάλληλο για εκπαίδευση σε περιβάλλον περιορισμένων πόρων (Colab L4).

3.5 Πειραματικός σχεδιασμός και υπερπαραμέτροι

Ο πειραματικός σχεδιασμός οργανώθηκε με στόχο τη συγκριτική αξιολόγηση των LoRA FP16 και QLoRA υπό συνθήκες περιορισμένων και ετερογενών υπολογιστικών πόρων. Η βασική λογική ήταν να διατηρούνται, όπου είναι εφικτό, κοινές ρυθμίσεις ανά μοντέλο και περιβάλλον, ώστε οι διαφορές στις μετρικές ποιότητας, χρόνου και μνήμης να αποδίδονται κυρίως στη μέθοδο προσαρμογής και όχι σε άσχετες μεταβολές της πειραματικής διαδικασίας.

Για το dataset_v1 ακολουθήθηκε παραγοντικός σχεδιασμός με δύο μοντέλα, δύο μεθόδους προσαρμογής και τρεις τιμές του LoRA rank. Συγκεκριμένα, αξιολογήθηκαν τα TinyLlama-1.1B και Gemma-2-2B με LoRA FP16 και QLoRA, για $r=8$, $r=16$ και $r=32$. Και στις δύο μεθόδους χρησιμοποιήθηκαν ως target modules τα `q_proj` και `v_proj`, ώστε η σύγκριση να παραμείνει ελεγχόμενη και να επικεντρωθεί στην επίδραση της ποσοτικοποίησης και του rank. Η εκπαίδευση πραγματοποιήθηκε για μία εποχή, επιλογή που συνδέεται με τον περιορισμένο υπολογιστικό προϋπολογισμό και την ανάγκη εκτέλεσης πολλών επαναλήψιμων πειραμάτων σε διαφορετικά περιβάλλοντα. Η επιλογή αυτή επιτρέπει την αποτύπωση αρχικών τάσεων ως προς μνήμη, χρόνο και συμπεριφορά σύγκλισης, αλλά δεν πρέπει να ερμηνευθεί ως πλήρης διερεύνηση της τελικής ποιότητας μετά από εκτεταμένη εκπαίδευση.

Το dataset_v1 χρησιμοποιήθηκε ως μικρό σύνολο αναφοράς για γρήγορα και συγκρίσιμα πειράματα. Η χρήση του επέτρεψε να εξεταστούν πολλαπλοί συνδυασμοί μοντέλων, μεθόδων και ranks χωρίς να απαιτηθεί υπερβολικός χρόνος εκπαίδευσης. Παράλληλα, το dataset_wiki_50k αξιοποιήθηκε για πειράματα μεγαλύτερης κλίμακας και ελέγχους γενίκευσης, ώστε να εξεταστεί αν οι τάσεις που παρατηρούνται στο μικρότερο σύνολο διατηρούνται και σε διαφορετικό ελληνικό corpus. Με αυτόν τον τρόπο, το dataset_v1 λειτουργεί ως ελεγχόμενο σημείο εκκίνησης, ενώ το dataset_wiki_50k προσθέτει ένα δεύτερο επίπεδο αξιολόγησης σε πιο ευρύ γλωσσικό υλικό.

Στο περιβάλλον Colab L4 χρησιμοποιήθηκαν ρυθμίσεις προσαρμοσμένες στους διαθέσιμους πόρους κάθε μοντέλου. Για το TinyLlama εφαρμόστηκε `max_length=256`, `batch size 1` και `gradient accumulation 8`, ενώ η αξιολόγηση βασίστηκε σε `eval_test_size=0.05` και `eval_max_samples=500`. Για το Gemma-2-2B, το οποίο έχει μεγαλύτερες απαιτήσεις μνήμης, η αρχική χρήση `max_length=256` οδήγησε σε σφάλματα Out-Of-Memory. Για τον λόγο αυτό, το `max_length` μειώθηκε σε 128 και η αξιολόγηση έγινε πιο ελαφριά, με `eval_test_size=0.02` και `eval_max_samples=100`. Η επιλογή αυτή δεν έγινε για να ευνοήσει κάποια μέθοδο, αλλά για να διασφαλιστεί ότι τα πειράματα Gemma-2-2B μπορούν να ολοκληρωθούν σταθερά στο ίδιο περιβάλλον και με συγκρίσιμες ρυθμίσεις μεταξύ LoRA FP16 και QLoRA.

Για το Gemma-2-2B χρησιμοποιήθηκαν επιπλέον ρυθμίσεις σταθερότητας, όπως `gradient accumulation 16`, απενεργοποίηση ορισμένων μηχανισμών `compilation` και ρυθμίσεις διαχείρισης μνήμης μέσω του PyTorch allocator. Επίσης, στις σχετικές εκτελέσεις

επιλέχθηκε `attn_implementation='eager'`, επειδή σε ορισμένους συνδυασμούς Gemma-2, 4-bit quantization και LoRA adapters μπορεί να εμφανιστούν ασυμβατότητες τύπων δεδομένων στη διαδρομή SDPA. Η επιλογή του eager attention είχε ως προτεραιότητα τη σταθερότητα και την αναπαραγωγικότητα των εκτελέσεων, αν και ενδέχεται να επηρεάζει τον χρόνο εκπαίδευσης σε σχέση με πιο βελτιστοποιημένα attention kernels. Για τον λόγο αυτό, οι χρονικές συγκρίσεις που αφορούν το Gemma-2-2B ερμηνεύονται με προσοχή.

Οι δύο μέθοδοι υλοποιήθηκαν με διαφορετικό τρόπο ως προς την αναπαράσταση των βαρών και τον optimizer. Στην QLoRA χρησιμοποιήθηκε φόρτωση της βάσης του μοντέλου σε 4-bit με NF4 quantization, double quantization και `paged_adamw_8bit` optimizer, ώστε να μειωθεί η χρήση VRAM και να περιοριστούν αιχμές μνήμης. Στη LoRA FP16, το βασικό μοντέλο φορτώθηκε σε fp16 και χρησιμοποιήθηκε `adamw_torch` optimizer. Και στις δύο περιπτώσεις εκπαιδεύτηκαν μόνο οι LoRA παράμετροι, ενώ το βασικό μοντέλο παρέμεινε παγωμένο.

Συνολικά, ο σχεδιασμός επιδιώκει να απομονώσει την πρακτική αντιστάθμιση ανάμεσα σε ποιότητα, χρόνο και μνήμη. Η εργασία δεν στοχεύει σε εξαντλητικό benchmark όλων των πιθανών υπερπαραμέτρων, αλλά σε μια αναπαραγωγίμη συγκριτική μελέτη που δείχνει ποιες επιλογές παραμένουν ρεαλιστικές όταν το fine-tuning μεγάλων γλωσσικών μοντέλων γίνεται σε ελληνικά δεδομένα και σε περιβάλλοντα με περιορισμένη ή μεταβαλλόμενη διαθέσιμη VRAM.

3.5.1 Άξονες πειραματικής αξιολόγησης

Η πειραματική αξιολόγηση οργανώνεται γύρω από τέσσερις βασικούς άξονες. Ο πρώτος αφορά την ποιότητα προσαρμογής και εξετάζει το eval loss, το perplexity και το token-level accuracy. Ο δεύτερος σχετίζεται με την αποδοτικότητα και αποτυπώνεται στον χρόνο εκπαίδευσης. Ο τρίτος αφορά τις απαιτήσεις πόρων και αποτιμάται μέσω της μέγιστης χρήσης μνήμης GPU (peak VRAM). Ο τέταρτος άξονας αφορά τη συμπεριφορά γενίκευσης, δηλαδή το πώς μεταφέρεται η προσαρμογή όταν η αξιολόγηση γίνεται σε διαφορετικό σύνολο δεδομένων από εκείνο της εκπαίδευσης. Η αξιολόγηση δεν περιορίζεται σκόπιμα σε μια σύγκριση της τελικής perplexity, επειδή κάτι τέτοιο θα άφηνε εκτός το κόστος χρόνου, τη μνήμη και τη συμπεριφορά γενίκευσης.

3.5.2 Στόχοι της πειραματικής αξιολόγησης

Με βάση τους παραπάνω άξονες, η εργασία επιδιώκει να δείξει πότε η QLoRA προσφέρει ουσιαστικό όφελος μνήμης χωρίς καθαρή απώλεια ποιότητας, πότε η LoRA FP16 παραμένει

προτιμότερη επειδή εκπαιδεύεται γρηγορότερα ή δίνει πιο σταθερή εικόνα στα αποτελέσματα, και πώς αλλάζει αυτή η σύγκριση όταν μεταβάλλεται το υπολογιστικό περιβάλλον, το μήκος ακολουθίας ή το ίδιο το σύνολο δεδομένων.

3.6 Μετρικές και αρχεία αποτελεσμάτων

Η αξιολόγηση των πειραμάτων βασίστηκε σε συνδυασμό μετρικών ποιότητας και αποδοτικότητας, ώστε η σύγκριση μεταξύ LoRA FP16 και QLoRA να μην περιοριστεί μόνο στην τελική επίδοση του μοντέλου, αλλά να λάβει υπόψη και το υπολογιστικό κόστος κάθε προσέγγισης. Για τον σκοπό αυτό, κάθε πείραμα παρήγαγε ένα κοινό σύνολο αρχείων αποτελεσμάτων, ώστε οι εκτελέσεις να μπορούν να συγκριθούν με ενιαίο και αναπαραγώγιμο τρόπο.

Η βασική μετρική ποιότητας ήταν το eval loss, όπως υπολογίζεται από τη διαδικασία αξιολόγησης του Trainer. Από το eval loss υπολογίστηκε το perplexity μέσω της σχέσης $\text{perplexity} = \exp(\text{eval_loss})$. Το perplexity χρησιμοποιείται ως βασική ένδειξη της ικανότητας του μοντέλου να προβλέπει το επόμενο token σε δεδομένα αξιολόγησης. Χαμηλότερες τιμές perplexity υποδηλώνουν καλύτερη πιθανοτική προσαρμογή στο συγκεκριμένο σύνολο αξιολόγησης, χωρίς όμως να σημαίνουν απαραίτητα καλύτερη συμπεριφορά σε όλες τις πιθανές εφαρμογές παραγωγής κειμένου.

Συμπληρωματικά υπολογίστηκε token-level accuracy, δηλαδή η ακρίβεια πρόβλεψης του επόμενου token στο eval set. Ο υπολογισμός έγινε συγκρίνοντας το token με τη μέγιστη προβλεπόμενη πιθανότητα με το πραγματικό επόμενο token, εξαιρώντας θέσεις που αντιστοιχούν σε padding ή έχουν label -100. Η μετρική αυτή χρησιμοποιήθηκε ως πρόσθετη ένδειξη σύγκρισης μεταξύ πειραμάτων, αλλά δεν αντιμετωπίζεται ως πλήρης αξιολόγηση ικανότητας του μοντέλου σε πραγματικές downstream εργασίες. Εργασίες όπως question answering, classification ή αξιολόγηση σημασιολογικής ποιότητας παραγόμενων απαντήσεων θα απαιτούσαν ξεχωριστό πρωτόκολλο αξιολόγησης και διαφορετικά σύνολα δεδομένων.

Για την αποτίμηση της αποδοτικότητας καταγράφηκαν ο χρόνος εκπαίδευσης και η μέγιστη χρήση μνήμης GPU. Ο χρόνος εκπαίδευσης αποθηκεύτηκε ως train_runtime_sec, ενώ η μέγιστη χρήση VRAM καταγράφηκε ως peak_vram_mib, με βάση τη μέγιστη μνήμη που ανέφερε το PyTorch κατά την εκτέλεση. Επιπλέον, όπου ήταν διαθέσιμο, αποθηκεύονταν ενδεικτικά GPU snapshots μέσω nvidia-smi πριν ή μετά την εκτέλεση. Οι μετρήσεις αυτές

είναι κρίσιμες για την παρούσα εργασία, επειδή ο βασικός στόχος δεν είναι μόνο να συγκριθεί η ποιότητα LoRA και QLoRA, αλλά και να εξεταστεί ποια μέθοδος είναι πρακτικά εφικτή σε περιβάλλοντα με περιορισμένη VRAM.

Κάθε εκτέλεση αποθήκευε τα αποτελέσματά της σε τυποποιημένη δομή φακέλων. Το αρχείο `metrics.json` περιείχε τις βασικές μετρικές του πειράματος, όπως `train_loss`, `eval_loss`, `perplexity`, `token_accuracy`, `train_runtime_sec`, `global_steps` και `peak_vram_mib`. Το αρχείο `results_table.csv` παρείχε την ίδια πληροφορία σε μορφή πίνακα, ώστε να μπορεί να χρησιμοποιηθεί για συγκεντρωτική ανάλυση και παραγωγή γραφημάτων. Το `log_history.csv` περιείχε το ιστορικό εκπαίδευσης, όπως καταγράφηκε από τον Trainer, και χρησιμοποιήθηκε για την παραγωγή καμπυλών `loss` όπου υπήρχαν επαρκή σημεία καταγραφής. Το `loss_curve.png` αποθήκευε την αντίστοιχη καμπύλη εκπαίδευσης, ενώ το `generations.json` περιείχε ενδεικτικές απαντήσεις του μοντέλου σε σταθερά ερωτήματα ποιοτικού ελέγχου.

Για λόγους αναπαραγωγιμότητας, κάθε πείραμα έλαβε μοναδικό αναγνωριστικό εκτέλεσης και `canonical run id` της μορφής `run_<model>_<dataset>_<method>_r<r>`. Η χρήση σταθερής ονοματοδοσίας επέτρεψε την αντιστοίχιση μετρικών, γραφημάτων, `adapters` και αρχείων καταγραφής ανά πείραμα. Παράλληλα, χρησιμοποιήθηκε σταθερό `seed`, ώστε να μειωθεί όσο είναι δυνατόν η τυχαιότητα μεταξύ εκτελέσεων. Παρότι η πλήρης ντετερμινιστικότητα δεν μπορεί να διασφαλιστεί σε όλα τα GPU περιβάλλοντα, η κοινή δομή αποτελεσμάτων καθιστά δυνατή τη συστηματική σύγκριση μεταξύ μοντέλων, μεθόδων, `ranks` και υπολογιστικών περιβαλλόντων.

Συνολικά, οι μετρικές που χρησιμοποιήθηκαν καλύπτουν τρεις βασικές διαστάσεις της αξιολόγησης: την ποιότητα προσαρμογής, την υπολογιστική αποδοτικότητα και την πρακτική εφικτότητα. Με αυτόν τον τρόπο, η αξιολόγηση ευθυγραμμίζεται με τον στόχο της εργασίας, δηλαδή τη σύγκριση τεχνικών προσαρμογής μεγάλων γλωσσικών μοντέλων όχι μόνο ως προς την τελική απόδοση, αλλά και ως προς το κόστος και τους περιορισμούς που εμφανίζονται σε πραγματικά πειραματικά περιβάλλοντα.

4. Αποτελέσματα (RTX 3070) και πλαίσια σύγκρισης με Colab

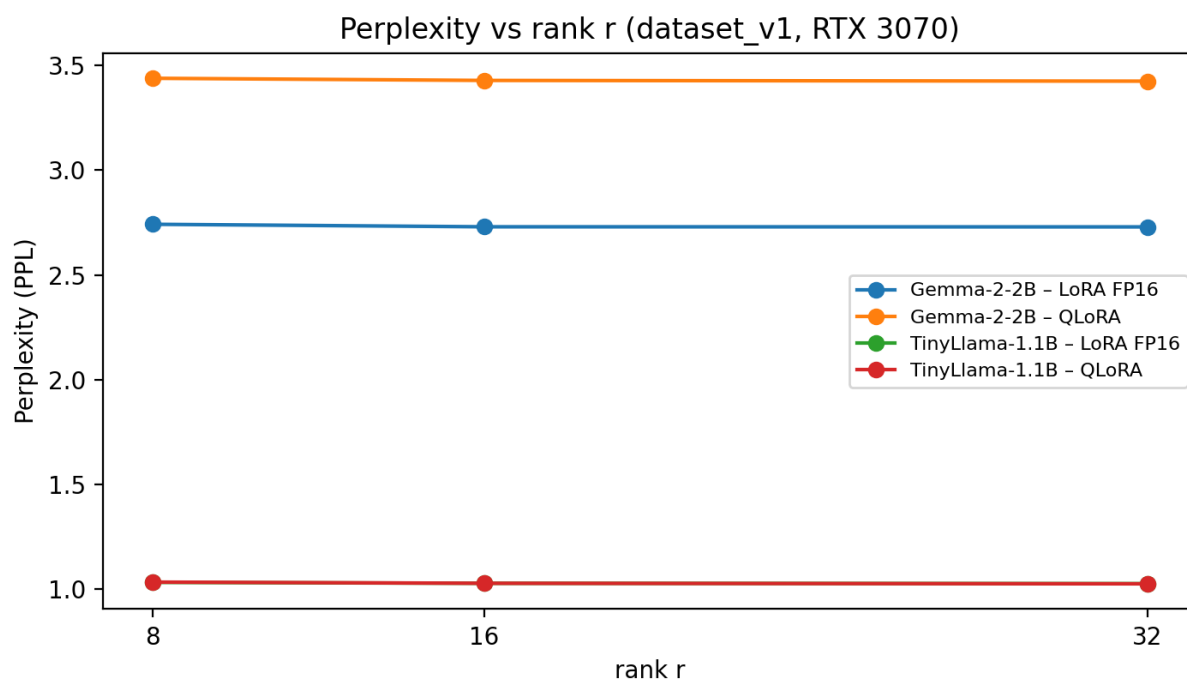
4.1 Σύννοψη πειραμάτων RTX 3070

Η ενότητα παρουσιάζει τα αποτελέσματα που προέκυψαν από εκτελέσεις σε NVIDIA GeForce RTX 3070. Τα πειράματα στο dataset_v1 περιλαμβάνουν TinyLlama και Gemma-2-2B με LoRA FP16 και QLoRA για $r=8/16/32$. Επιπλέον, στο dataset_wiki_50k εκτελέστηκαν πειράματα κλιμάκωσης ($r=16$, QLoRA).

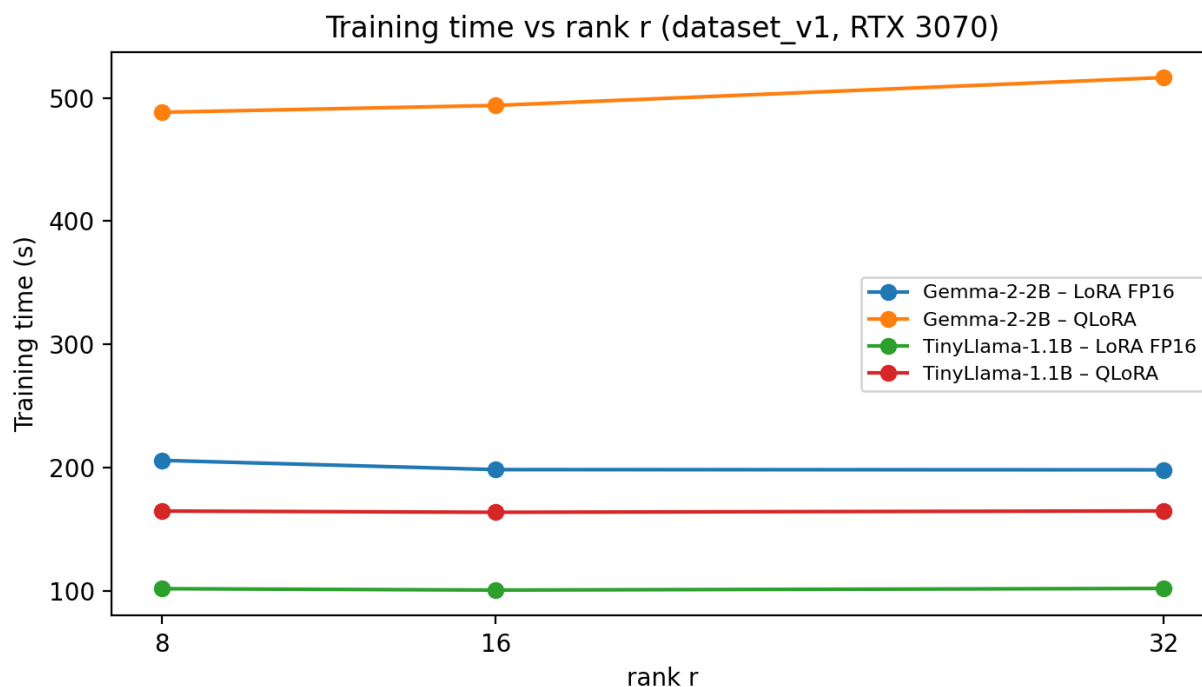
Model	Method	Dataset	r	PPL	eval loss	token acc	time (s)	VRAM (MiB)	run_id
Gemma-2-2B	LoRA FP16	dataset_v1	8	2.742	1.009	0.923	205.7	12033	run_gemma2_2b_v1_lora_fp16_r8
Gemma-2-2B	LoRA FP16	dataset_v1	16	2.730	1.004	0.932	198.2	12054	run_gemma2_2b_v1_lora_fp16_r16
Gemma-2-2B	LoRA FP16	dataset_v1	32	2.730	1.004	0.932	198.1	12097	run_gemma2_2b_v1_lora_fp16_r32
Gemma-2-2B	QLoRA	dataset_v1	8	3.439	1.235	0.923	488.5	10316	run_gemma2_2b_v1_qlora_r8
Gemma-2-2B	QLoRA	dataset_v1	16	3.429	1.232	0.928	494.0	10330	run_gemma2_2b_v1_qlora_r16
Gemma-2-2B	QLoRA	dataset_v1	32	3.425	1.231	0.932	516.7	10357	run_gemma2_2b_v1_qlora_r32
TinyLlama-1.1B	LoRA FP16	dataset_v1	8	1.033	0.032	0.987	101.6	3014	run_tinyllama_v1_lora_fp16_r8
TinyLlama-1.1B	LoRA FP16	dataset_v1	16	1.029	0.028	0.987	100.4	3029	run_tinyllama_v1_lora_fp16_r16
TinyLlama-1.1B	LoRA FP16	dataset_v1	32	1.027	0.027	0.985	101.8	3059	run_tinyllama_v1_lora_fp16_r32
TinyLlama-1.1B	QLoRA	dataset_v1	8	1.034	0.034	0.985	164.6	1905	run_tinyllama_v1_qlora_r8
TinyLlama-1.1B	QLoRA	dataset_v1	16	1.029	0.028	0.987	163.6	1911	run_tinyllama_v1_qlora_r16
TinyLlama-1.1B	QLoRA	dataset_v1	32	1.026	0.026	0.987	164.7	1930	run_tinyllama_v1_qlora_r32
Gemma-2-2B	QLoRA	dataset_wiki_50k	16	6.128	1.813	0.582	18831.2	10328	run_gemma2_2b_wiki50k_qlora_r16
TinyLlama-1.1B	QLoRA	dataset_wiki_50k	16	3.111	1.135	0.669	7630.7	1908	run_tinyllama_wiki50k_qlora_r16

Πίνακας 4.1: Συνοπτικά αποτελέσματα RTX 3070 (dataset_v1 και dataset_wiki_50k).

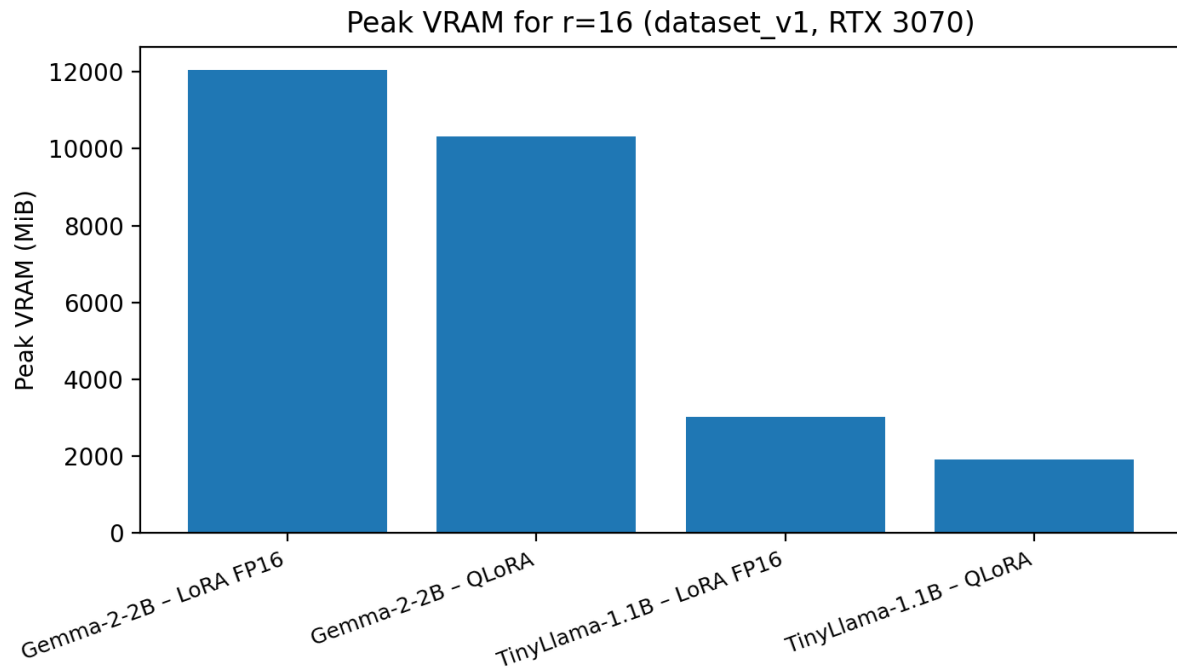
4.2 Γραφήματα (RTX 3070)



Σχήμα 4.1: Perplexity ως συνάρτηση του rank r (dataset_v1, RTX 3070).

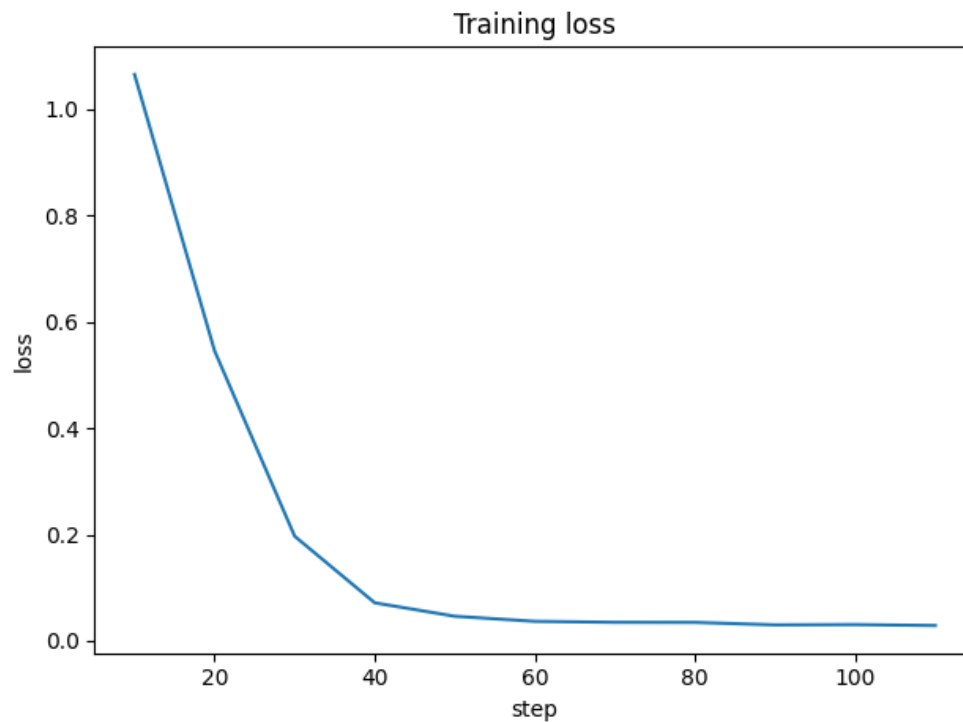


Σχήμα 4.2: Χρόνος εκπαίδευσης ως συνάρτηση του rank r (dataset_v1, RTX 3070).

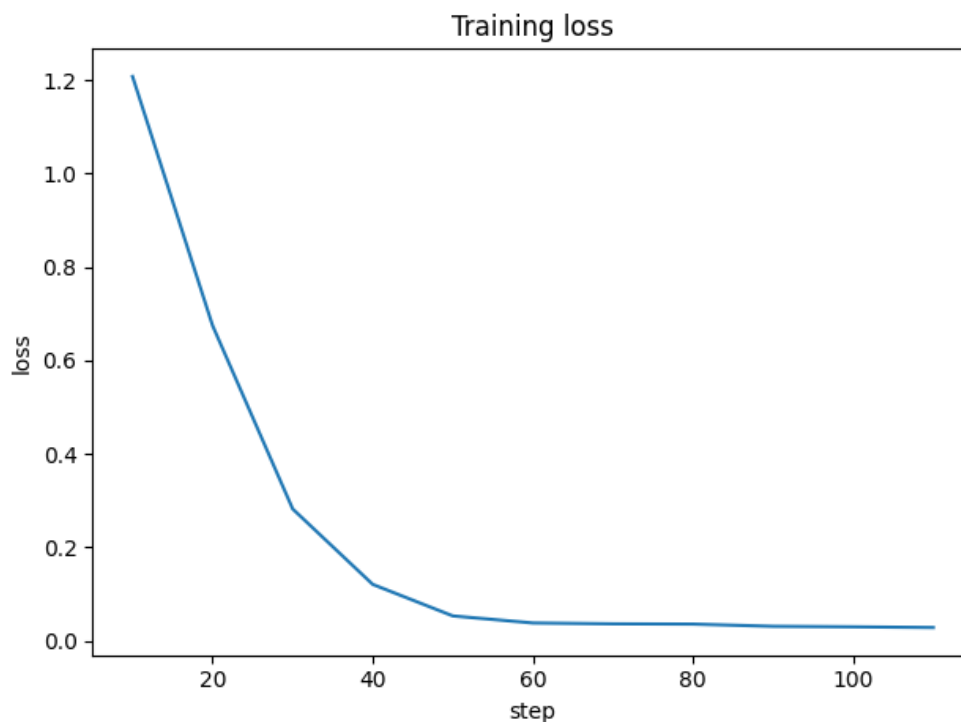


Σχήμα 4.3: Μέγιστη χρήση VRAM για r=16 (dataset_v1, RTX 3070).

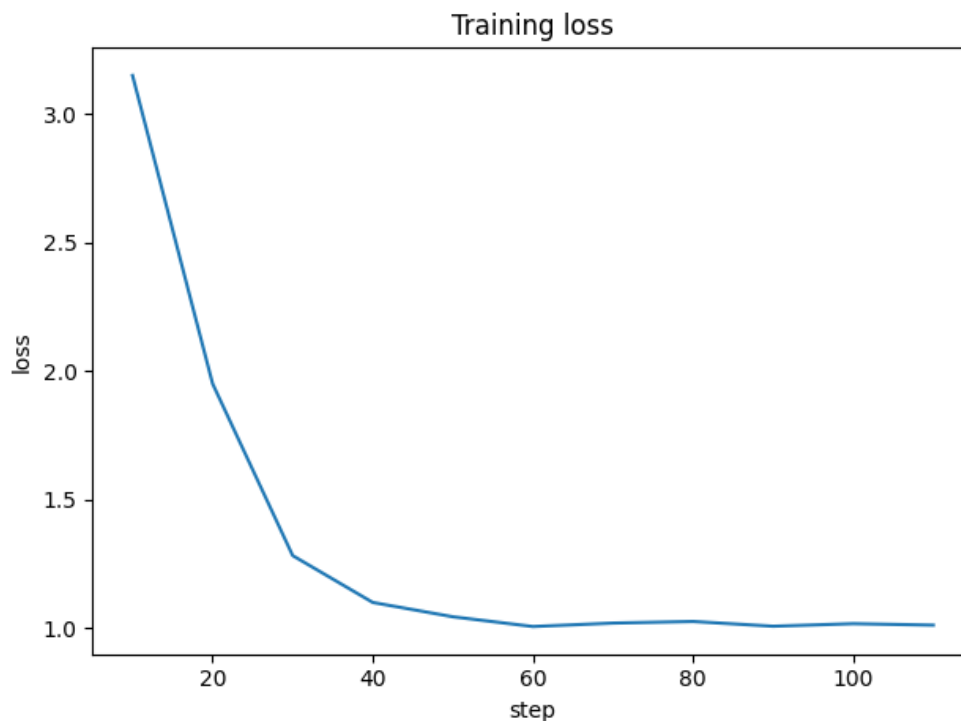
4.3 Καμπύλες σύγκλισης (Loss Curves, r=16)



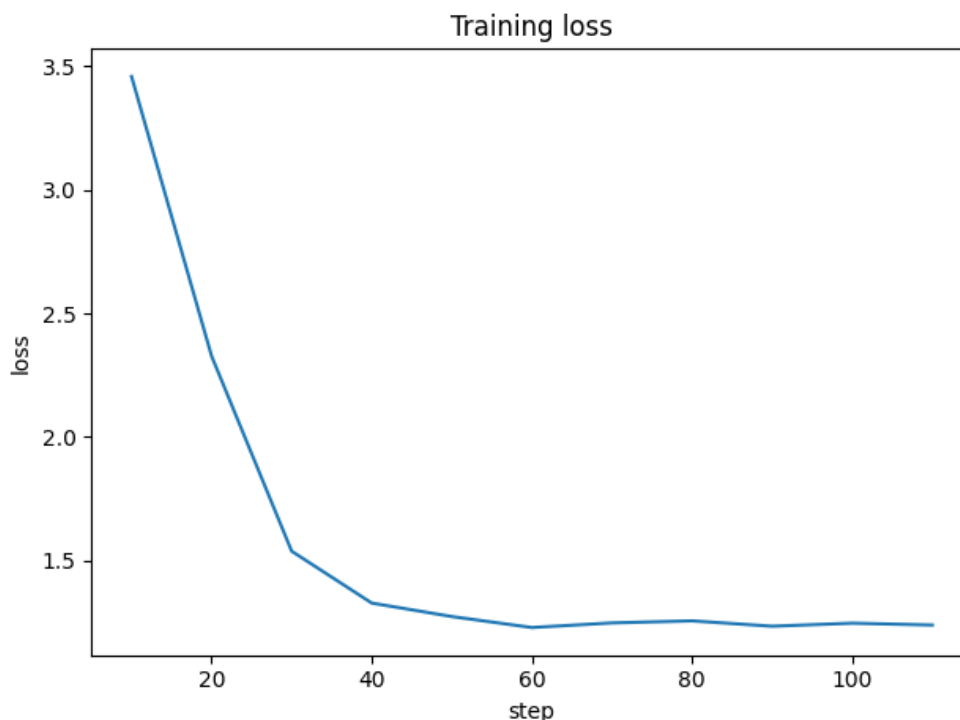
Σχήμα 4.4: TinyLlama – LoRA FP16 (r=16): καμπύλη training loss.



Σχήμα 4.5: TinyLlama – QLoRA (r=16): καμπύλη training loss.



Σχήμα 4.6: Gemma-2-2B – LoRA FP16 (r=16): καμπύλη training loss.



Σχήμα 4.7: Gemma-2-2B – QLoRA (r=16): καμπύλη training loss.

4.4 Παρατηρήσεις από τα αποτελέσματα RTX 3070

TinyLlama-1.1B (r=16, dataset_v1): Η LoRA FP16 πέτυχε perplexity 1.029, ενώ η QLoRA 1.029 (διαφορά +0.0%). Σε χρόνο εκπαίδευσης, FP16 100.4s έναντι QLoRA 163.6s (+62.9%). Σε peak VRAM, FP16 3029 MiB έναντι QLoRA 1911 MiB (-36.9%).

Gemma-2-2B (r=16, dataset_v1): Η LoRA FP16 πέτυχε perplexity 2.730, ενώ η QLoRA 3.429 (διαφορά +25.6%). Σε χρόνο εκπαίδευσης, FP16 198.2s έναντι QLoRA 494.0s (+149.2%). Σε peak VRAM, FP16 12054 MiB έναντι QLoRA 10330 MiB (-14.3%).

4.5 Πλαίσιο ενοποίησης για σύγκριση με Colab

Τα αποτελέσματα από το Colab (GPU L4 και GPU A100) εξήχθησαν με κοινή δομή ανά πείραμα (metrics.json, results_table.csv, log_history.csv, loss_curve.png, generations.json) και ενσωματώθηκαν στους Πίνακες 4.2 και 4.3. Η κοινή μορφή έκανε δυνατή την αντιπαραβολή τους με τα αντίστοιχα τοπικά αποτελέσματα, χωρίς να αλλάξει ο τρόπος καταγραφής των μετρήσεων.

4.6 Ρυθμίσεις πειραμάτων Colab (GPU L4)

Στο Colab τα πειράματα εκτελούνται μέσω σεναρίων που καλούν τον ίδιο κορμό εκπαίδευσης και αξιολόγησης, ώστε να διατηρείται κοινή μορφή αρχείων αποτελεσμάτων και κοινή ονοματοδοσία. Για το TinyLlama χρησιμοποιείται το πλήρες αρχικό πείραμα αναφοράς (max_length=256, eval_max_samples=500), ενώ για το Gemma2 εφαρμόζονται πιο συντηρητικές ρυθμίσεις (max_length=128, ελαφρύτερη αξιολόγηση) λόγω των περιορισμών μνήμης στο L4.

Για Gemma2 έχουν προστεθεί επιπλέον ρυθμίσεις σταθερότητας για επαναλαμβανόμενες εκτελέσεις (ρυθμίσεις allocator για κατακερματισμό μνήμης και απενεργοποίηση torch compile/dynamo), καθώς και attn_implementation='eager' για συμβατότητα.

4.7 Πίνακας αποτελεσμάτων Colab (GPU L4)

Ο ακόλουθος πίνακας συμπληρώθηκε από τα αποτελέσματα Colab L4 (12 πειράματα: 2 μοντέλα × 2 μέθοδοι × 3 τιμές rank) και χρησιμοποιείται στην ανάλυση LoRA έναντι QLoRA στο περιβάλλον L4.

Env	Model	Method	r	L	train loss	eval loss	PPL	token acc	VRAM (MiB)
Colab L4	Gemma-2-2B	LoRA FP16	8	128	0.924	0.266	1.305	0.949	8529
Colab L4	Gemma-2-2B	LoRA FP16	16	128	0.581	0.094	1.099	0.963	8550
Colab L4	Gemma-2-2B	LoRA FP16	32	128	0.393	0.061	1.063	0.967	8593
Colab L4	Gemma-2-2B	QLoRA	8	128	0.902	0.231	1.260	0.949	6785
Colab L4	Gemma-2-2B	QLoRA	16	128	0.568	0.090	1.094	0.969	6797
Colab L4	Gemma-2-2B	QLoRA	32	128	0.390	0.061	1.063	0.963	6822
Colab L4	TinyLlama-1.1B	LoRA FP16	8	256	0.246	0.033	1.034	0.987	3013
Colab L4	TinyLlama-1.1B	LoRA FP16	16	256	0.182	0.029	1.029	0.987	3028
Colab L4	TinyLlama-1.1B	LoRA FP16	32	256	0.134	0.027	1.027	0.985	3058
Colab L4	TinyLlama-1.1B	QLoRA	8	256	0.300	0.034	1.035	0.987	1894
Colab L4	TinyLlama-1.1B	QLoRA	16	256	0.217	0.029	1.029	0.987	1902

Env	Model	Method	r	L	train loss	eval loss	PPL	token acc	VRAM (MiB)
Colab L4	TinyLlama-1.1B	QLoRA	32	256	0.160	0.026	1.026	0.987	1919

Πίνακας 4.2: Συνοπτικά αποτελέσματα Colab L4 (dataset_v1).

Παρατήρηση μετρήσεων: στις πειραματικές εκτελέσεις χρησιμοποιήθηκε σταθερό padding στο max_length (padding="max_length") για λόγους συγκρισιμότητας. Ο υπολογισμός των eval loss και perplexity έγινε με συνεπή χειρισμό των padding tokens (masking σε -100 όπου απαιτείται), ώστε οι τιμές να μπορούν να συγκριθούν με τον ίδιο τρόπο μεταξύ περιβαλλόντων.

4.8 Σύγκριση Colab L4 έναντι RTX 3070 (πλαίσιο ερμηνείας)

Τα πειράματα αναπαραγωγής των Πινάκων 4.1 και 4.2 δείχνουν καθαρά το γνώριμο trade-off ανάμεσα σε μνήμη και χρόνο. Στο TinyLlama-1.1B (r=16, max_length=256) η QLoRA μείωσε την peak VRAM κατά περίπου 37.2% (από 3028 σε 1902 MiB), αλλά αύξησε τον χρόνο εκπαίδευσης κατά περίπου 88.9% (από 153.9 σε 290.7 sec). Στο Gemma-2-2B (max_length=128 στο L4), η μείωση μνήμης ήταν περίπου 20.5%, με αντίστοιχη αύξηση χρόνου περίπου 59.3%.

Για να είναι συγκρίσιμα τα αποτελέσματα στο Gemma-2-2B μεταξύ περιβαλλόντων, εκτελέστηκαν επιπλέον τοπικά πειράματα υπό ίδιες υπερπαραμέτρους όπως στο Colab (r=16, max_length=128, ίδιο split). Ο Πίνακας 4.3 συνοψίζει τη σύγκριση.

Env	Method	L	eval loss	PPL	token acc	time (s)	VRAM (MiB)
RTX 3070 (local)	QLoRA	128	1.250	3.491	0.934	266.8	6820
RTX 3070 (local)	LoRA FP16	128	1.026	2.789	0.934	140.9	8550
Colab L4	QLoRA	128	0.090	1.094	0.969	399.7	6797
Colab L4	LoRA FP16	128	0.094	1.099	0.963	250.9	8550
Colab A100	QLoRA	128	0.090	1.094	0.969	401.7	6797
Colab A100	LoRA FP16	128	0.094	1.099	0.963	249.7	8550

Πίνακας 4.3: Σύγκριση Gemma-2-2B υπό ίδιες ρυθμίσεις (r=16, max_length=128) σε RTX 3070, Colab L4 και Colab A100.

Οι χρόνοι στο RTX 3070 αποδείχθηκαν μικρότεροι από τους αντίστοιχους στο Colab L4/A100 για τα συγκεκριμένα πειράματα, ενώ η QLoRA διατήρησε σαφές πλεονέκτημα ως προς τη μνήμη (peak VRAM) έναντι της LoRA FP16. Οι αποκλίσεις στα μετρικά εντός πεδίου μεταξύ περιβαλλόντων δείχνουν ότι οι εκδόσεις βιβλιοθηκών, οι kernels και γενικότερα οι λεπτομέρειες εκτέλεσης μπορούν να επηρεάσουν την αναπαραγωγή. Για τον λόγο αυτό, τα κύρια συμπεράσματα στηρίζονται κυρίως σε πειράματα αναπαραγωγής ανά περιβάλλον και σε συγκρίσεις υπό κοινές υπερπαραμέτρους.

4.9 Πίνακας αποτελεσμάτων Colab (GPU A100)

Env	Model	Method	r	L	train loss	eval loss	PPL	token acc	VRAM (MiB)
Colab A100	Gemma-2-2B	LoRA FP16	8	128	0.924	0.266	1.305	0.949	8529
Colab A100	Gemma-2-2B	LoRA FP16	16	128	0.581	0.094	1.099	0.963	8550
Colab A100	Gemma-2-2B	LoRA FP16	32	128	0.393	0.061	1.063	0.967	8593
Colab A100	Gemma-2-2B	QLoRA	8	128	0.902	0.231	1.260	0.949	6785
Colab A100	Gemma-2-2B	QLoRA	16	128	0.568	0.090	1.094	0.969	6797
Colab A100	Gemma-2-2B	QLoRA	32	128	0.390	0.061	1.063	0.963	6822
Colab A100	TinyLlama-1.1B	LoRA FP16	8	256	0.245	0.033	1.033	0.987	3016
Colab A100	TinyLlama-1.1B	LoRA FP16	16	256	0.181	0.028	1.029	0.987	3031
Colab A100	TinyLlama-1.1B	LoRA FP16	32	256	0.133	0.027	1.027	0.985	3061
Colab A100	TinyLlama-1.1B	QLoRA	8	256	0.300	0.034	1.035	0.987	1901
Colab A100	TinyLlama-1.1B	QLoRA	16	256	0.217	0.029	1.029	0.987	1909
Colab A100	TinyLlama-1.1B	QLoRA	32	256	0.160	0.026	1.027	0.987	1927

Πίνακας 4.4: Συνοπτικά αποτελέσματα Colab A100 (dataset_v1).

4.10 Σύγκριση Colab L4 έναντι Colab A100 (αναπαραγωγή πειραμάτων)

Οι εκτελέσεις αναπαραγωγής στο A100 έγιναν με τις ίδιες υπερπαραμέτρους όπως στο L4, ώστε η σύγκριση να παραμείνει άμεση. Για $r=16$, στο TinyLlama καταγράφεται μια μικρή επιτάχυνση στο A100 (LoRA FP16 περίπου 2.9% και QLoRA περίπου 2.7% ταχύτερα ως προς τον χρόνο εκπαίδευσης), ενώ στο Gemma-2-2B οι χρόνοι είναι πρακτικά ίδιοι. Η γενική εικόνα, επομένως, δεν αλλάζει: η QLoRA μειώνει την peak VRAM σε σχέση με τη LoRA

FP16, με τίμημα μεγαλύτερο χρόνο εκπαίδευσης, ενώ το A100 επιτρέπει και ρυθμίσεις που δεν χωρούν εύκολα σε L4.

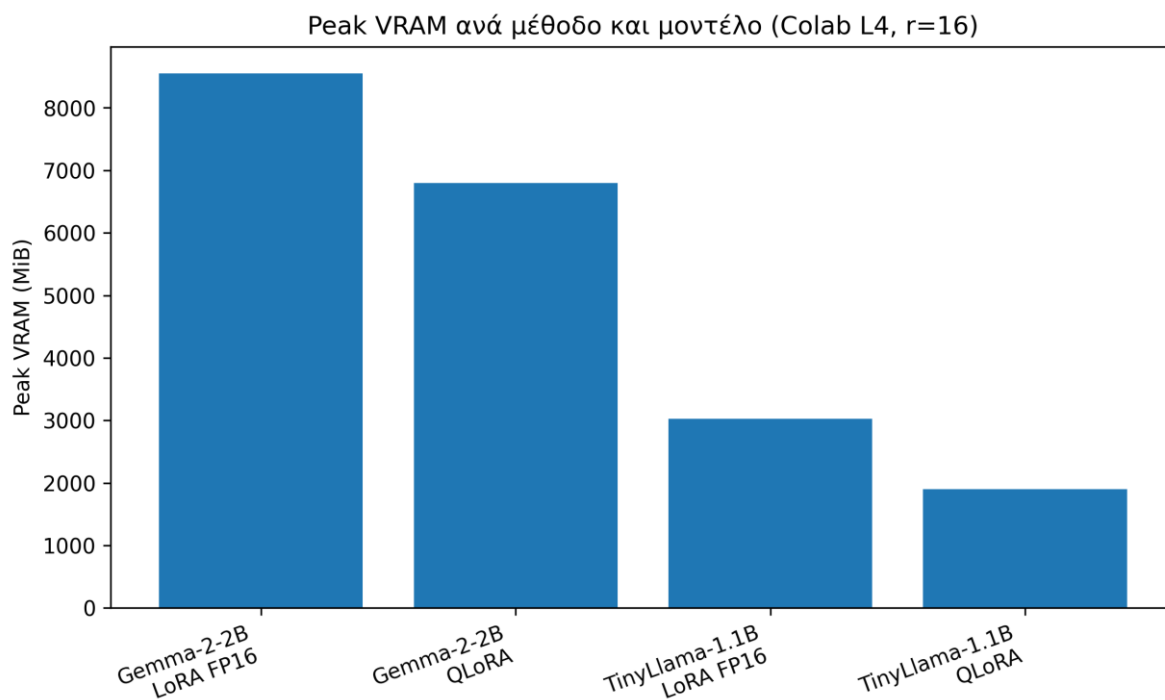
4.11 Συγκεντρωτική οπτικοποίηση απόδοσης και κόστους

Πέρα από τους αναλυτικούς πίνακες αποτελεσμάτων, κρίθηκε απαραίτητη η συγκεντρωτική οπτικοποίηση των βασικών μετρικών, ώστε να αποτυπωθεί καθαρότερα η σχέση ανάμεσα σε ποιότητα, χρόνο εκπαίδευσης και χρήση μνήμης. Τα γραφήματα που ακολουθούν λειτουργούν συμπληρωματικά προς τους πίνακες και αναδεικνύουν τα βασικά trade-offs μεταξύ LoRA FP16 και QLoRA.

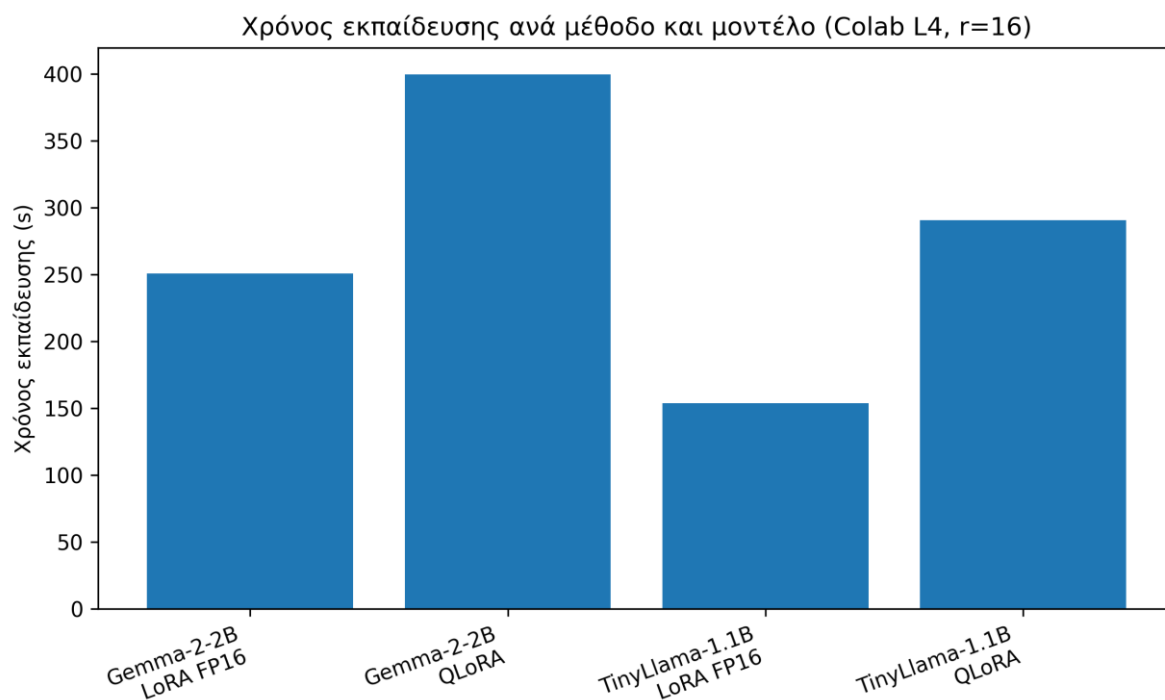
Η πρώτη ομάδα γραφημάτων συγκρίνει τις δύο μεθόδους ως προς τη μέγιστη χρήση VRAM, τον χρόνο εκπαίδευσης και το perplexity. Με αυτόν τον τρόπο αποτυπώνεται η βασική αντιστάθμιση που διατρέχει την εργασία: η QLoRA μειώνει τη χρήση μνήμης, αλλά συχνά απαιτεί μεγαλύτερο χρόνο εκπαίδευσης, ενώ η επίδρασή της στην τελική ποιότητα εξαρτάται από το μοντέλο, το περιβάλλον εκτέλεσης και τις επιμέρους υπερπαραμέτρους.

Η δεύτερη ομάδα γραφημάτων παρουσιάζει τα αποτελέσματα ως σχέση κόστους και ποιότητας. Η απεικόνιση VRAM–perplexity δείχνει πόση μνήμη απαιτείται για το επίπεδο ποιότητας που επιτυγχάνει κάθε εκτέλεση, ενώ η απεικόνιση χρόνου–perplexity δείχνει την αντίστοιχη σχέση ανάμεσα στη διάρκεια εκπαίδευσης και την τελική perplexity. Τα γραφήματα αυτά είναι ιδιαίτερα χρήσιμα για την ερμηνεία της πρακτικής αξίας κάθε μεθόδου, επειδή μετατοπίζουν την ανάλυση από το ερώτημα «ποια μέθοδος έχει καλύτερη perplexity» στο ερώτημα «ποια μέθοδος πετυχαίνει αποδεκτή ποιότητα με δεδομένους περιορισμούς μνήμης και χρόνου».

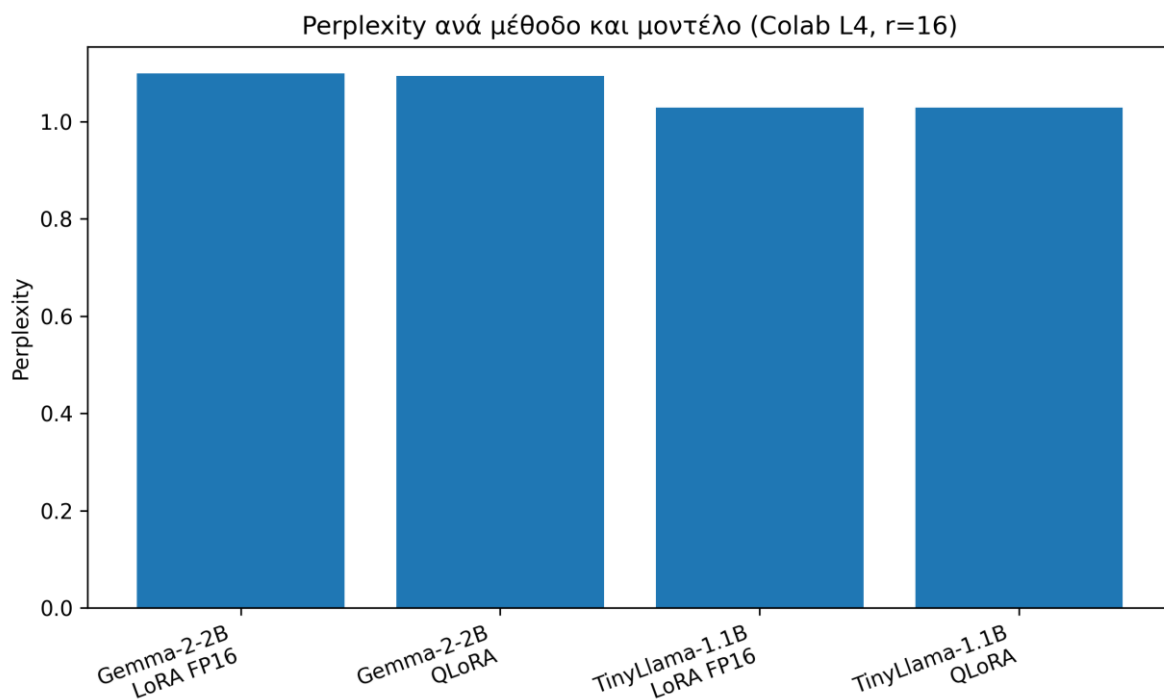
Τέλος, τα γραφήματα γενίκευσης και DAPT χρησιμοποιούνται για να αποτυπωθεί η συμπεριφορά των μοντέλων σε διαφορετικό corpus από εκείνο της εκπαίδευσης. Η οπτικοποίηση αυτή δείχνει ότι η αξιολόγηση δεν περιορίζεται στο in-domain αποτέλεσμα, αλλά εξετάζει και τη μεταφορά της προσαρμογής σε άλλο ελληνικό υλικό.



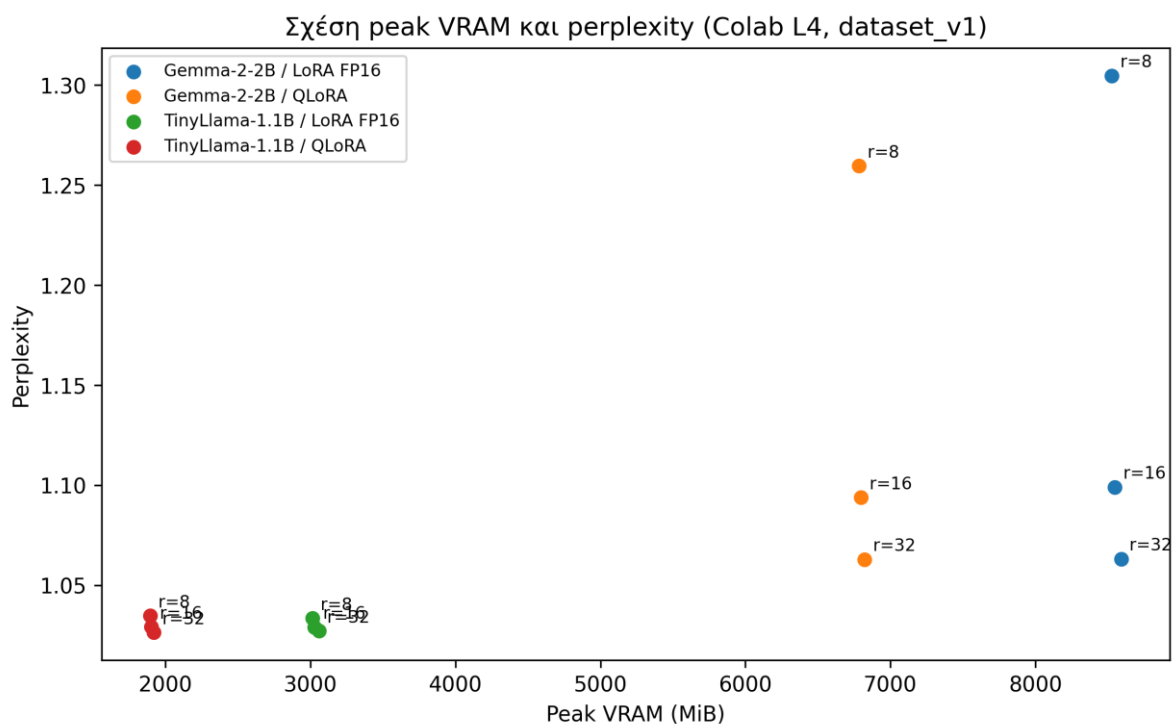
Σχήμα 4.8: Μέγιστη χρήση VRAM ανά μέθοδο και μοντέλο.



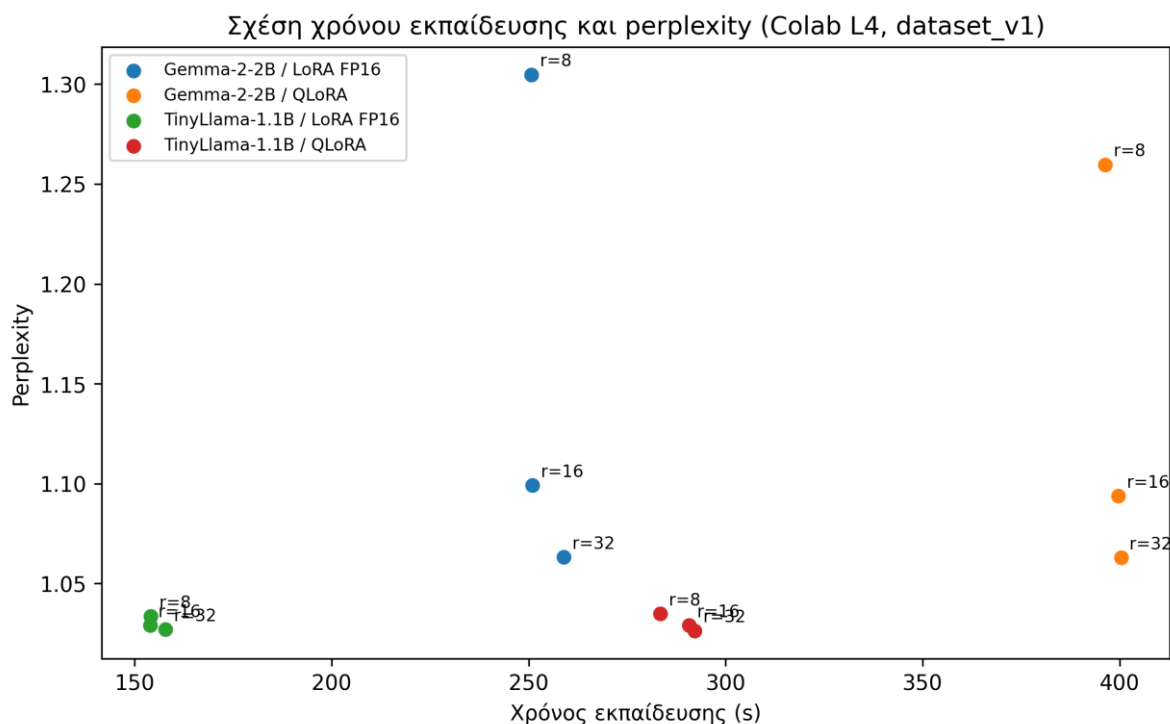
Σχήμα 4.9: Χρόνος εκπαίδευσης ανά μέθοδο και μοντέλο.



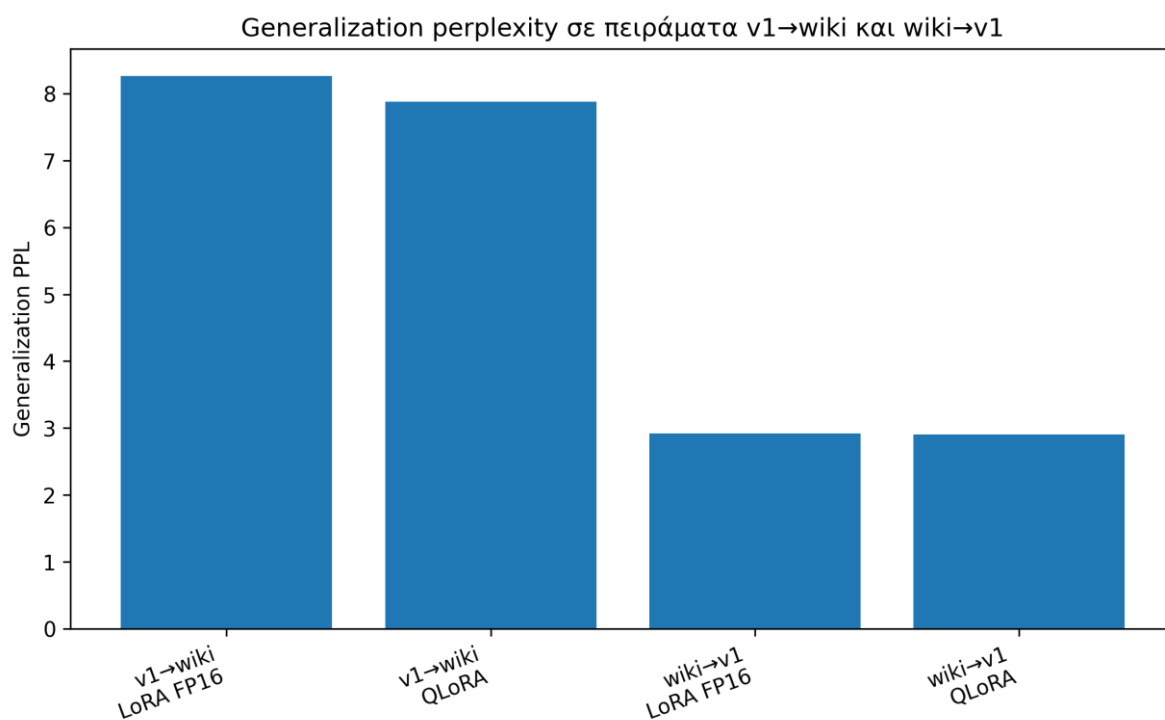
Σχήμα 4.10: Perplexity ανά μέθοδο και μοντέλο.



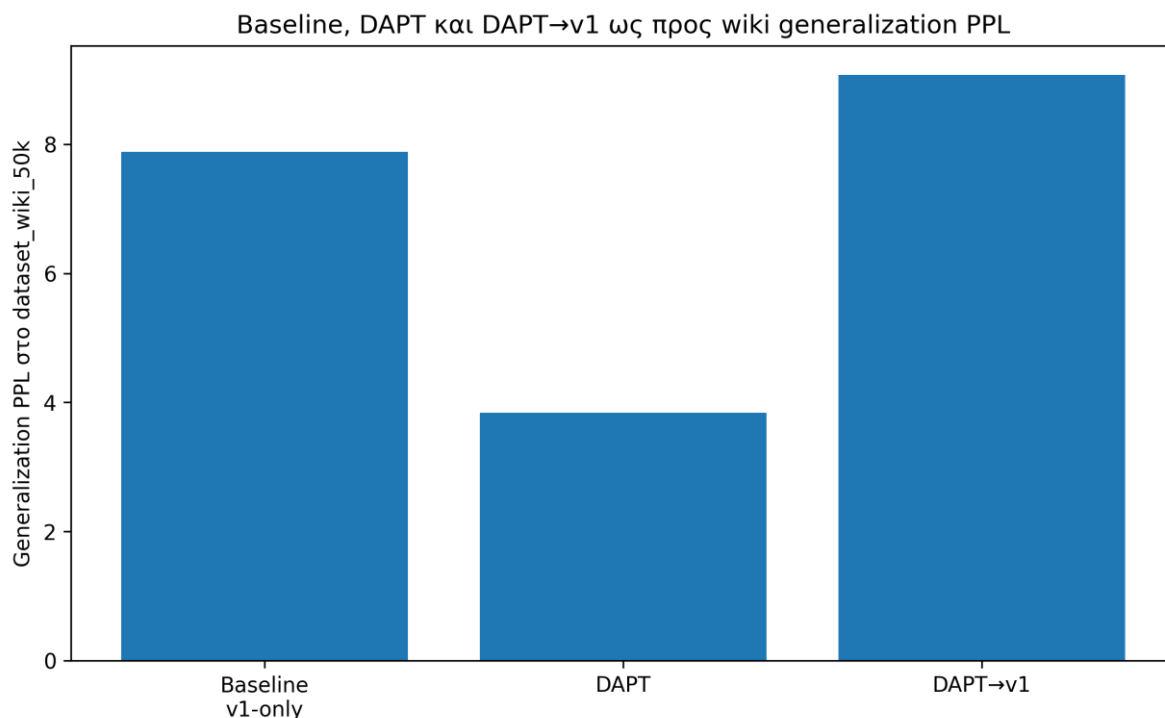
Σχήμα 4.11: Σχέση peak VRAM και perplexity.



Σχήμα 4.12: Σχέση χρόνου εκπαίδευσης και perplexity.



Σχήμα 4.13: Generalization perplexity σε πειράματα v1→wiki και wiki→v1.



Σχήμα 4.14: Σύγκριση baseline fine-tuning, DAPT και DAPT→fine-tuning ως προς wiki generalization perplexity.

4.12 Διερεύνηση ορίων στο Colab (GPU A100)

Πέρα από τα πειράματα αναπαραγωγής, πραγματοποιήθηκαν επιπλέον εκτελέσεις στο Colab με GPU A100, ώστε να εξεταστεί τι περιθώρια δίνει ένα ισχυρότερο υπολογιστικό περιβάλλον. Τα πειράματα αυτά δεν σχεδιάστηκαν ως άμεση σύγκριση με όλες τις προηγούμενες εκτελέσεις, αλλά ως διερεύνηση δυνατοτήτων. Για τον λόγο αυτό, σε ορισμένες περιπτώσεις μεταβλήθηκαν παράμετροι όπως το μήκος ακολουθίας, το batch size, το gradient accumulation και το εύρος των target modules.

Η πρώτη κατεύθυνση διερεύνησης αφορούσε την αύξηση του μήκους ακολουθίας. Στα βασικά πειράματα του Gemma-2-2B στο Colab L4 χρησιμοποιήθηκε max_length=128, κυρίως για λόγους αποφυγής σφαλμάτων μνήμης. Στο A100 δοκιμάστηκε μεγαλύτερο max_length=512 τόσο με LoRA FP16 όσο και με QLoRA, ενώ πραγματοποιήθηκε και δοκιμή καταπόνησης με max_length=1024 για QLoRA. Οι εκτελέσεις αυτές δείχνουν ότι το μεγαλύτερο περιθώριο VRAM επιτρέπει την επεξεργασία μεγαλύτερων συμφραζομένων, αλλά με σημαντική αύξηση στη χρήση μνήμης. Επομένως, το μήκος ακολουθίας αποτελεί κρίσιμη παράμετρο όταν ο στόχος είναι η μετάβαση από μικρά πειράματα εφικτότητας σε πιο απαιτητικά σενάρια εκπαίδευσης.

Η δεύτερη κατεύθυνση αφορούσε την αύξηση της διεκπεραιωτικής ικανότητας μέσω μεγαλύτερου batch size. Συγκεκριμένα, δοκιμάστηκε batch=2 με grad_accum=8, διατηρώντας το ίδιο effective batch size με τη βασική ρύθμιση batch=1 και grad_accum=16. Η ρύθμιση αυτή είχε στόχο να εξετάσει αν η καλύτερη αξιοποίηση της GPU μπορεί να μειώσει τον χρόνο εκπαίδευσης χωρίς να αλλάξει ουσιαστικά το συνολικό μέγεθος του batch που βλέπει η διαδικασία βελτιστοποίησης. Τα αποτελέσματα δείχνουν ότι η αύξηση του batch size μπορεί να βελτιώσει σημαντικά τον χρόνο εκτέλεσης, όταν υπάρχει αρκετή διαθέσιμη μνήμη.

Η τρίτη κατεύθυνση διερεύνησης αφορούσε την επέκταση των target modules στη LoRA FP16. Εκτός από τα q_proj και v_proj, δοκιμάστηκε ευρύτερο σύνολο modules που περιλαμβάνει επιπλέον attention και MLP προβολές. Η επιλογή αυτή αυξάνει την εκφραστικότητα του adapter, καθώς περισσότερα τμήματα του μοντέλου αποκτούν εκπαιδευσιμες low-rank ενημερώσεις. Ωστόσο, το όφελος σε eval loss συνοδεύεται από μεγαλύτερο χρόνο εκπαίδευσης και ελαφρώς αυξημένη χρήση μνήμης. Συνεπώς, η διεύρυνση των target modules αποτελεί χρήσιμη επιλογή όταν ο στόχος είναι η βελτίωση της ποιότητας, αλλά δεν είναι απαραίτητα η πιο αποδοτική λύση σε περιβάλλοντα περιορισμένων πόρων.

Τα αποτελέσματα συνοψίζονται στον Πίνακα 4.5. Από αυτά προκύπτει ότι το A100 δεν αλλάζει τη βασική σχέση ανάμεσα σε LoRA FP16 και QLoRA, αλλά επιτρέπει την εξέταση πιο απαιτητικών ρυθμίσεων που δεν είναι εύκολα εφικτές σε GPU χαμηλότερης μνήμης. Η αύξηση του batch size μειώνει αισθητά τον χρόνο εκπαίδευσης, ενώ η αύξηση του max_length οδηγεί σε έντονη αύξηση της peak VRAM. Η δοκιμή με max_length=1024 και QLoRA ολοκληρώθηκε επιτυχώς, αλλά με κατανάλωση μνήμης που ξεπερνά κατά πολύ τις δυνατότητες μιας καταναλωτικής GPU 8GB. Αυτό επιβεβαιώνει ότι η επιλογή υπερπαραμέτρων δεν μπορεί να αποσυνδεθεί από το διαθέσιμο υλικό και ότι οι τεχνικές PEFT πρέπει να αξιολογούνται πάντα σε σχέση με τους πραγματικούς περιορισμούς του περιβάλλοντος εκτέλεσης.

Run	Method	r	L	batch	accum	target modules	eval loss	PPL	token acc	time (s)	VRAM (MiB)
Batch=2	LoRA FP16	16	128	2	8	q_proj,v_proj	0.0948	1.0995	0.9634	131.1	8551
Batch=2	QLoRA	16	128	2	8	q_proj,v_proj	0.0903	1.0945	0.9686	201.4	6847
Len=512	LoRA FP16	16	512	1	16	q_proj,v_proj	0.1018	1.1072	0.9630	257.0	19065
Len=512	QLoRA	16	512	1	16	q_proj,v_proj	0.0987	1.1037	0.9630	400.0	17374

Run	Method	r	L	batch	accum	target modules	eval loss	PPL	token acc	time (s)	VRAM (MiB)
Wide modules	LoRA FP16	16	128	1	16	q_proj,k_proj,v_proj,o_proj, gate_proj,up_proj,down_proj	0.0564	1.0581	0.9651	389.7	8786
Len=1024	QLoRA	16	1024	1	16	q_proj,v_proj	0.0982	1.1032	0.9630	399.3	31411

Πίνακας 4.5: Αποτελέσματα διερεύνησης ορίων στο A100 (Gemma-2-2B, dataset_v1).

Συνοψίζοντας τα αποτελέσματα του Πίνακα 4.5, προκύπτουν τέσσερα βασικά σημεία. Πρώτον, με σταθερό max_length=128, η αύξηση του batch size σε 2 και η αντίστοιχη μείωση του grad_accum σε 8 μείωσαν τον χρόνο εκπαίδευσης περίπου κατά 47.5% για τη LoRA FP16 και 49.9% για την QLoRA, σε σχέση με τη βασική ρύθμιση batch=1 και grad_accum=16. Δεύτερον, η αύξηση του max_length σε 512 οδήγησε σε σημαντική αύξηση της peak VRAM, επιβεβαιώνοντας ότι το μήκος ακολουθίας αποτελεί κρίσιμη παράμετρο μνήμης. Τρίτον, η δοκιμή καταπόνησης με max_length=1024 και QLoRA ολοκληρώθηκε επιτυχώς, αλλά απαιτήσε περίπου 30.7 GiB VRAM, τιμή που υπερβαίνει κατά πολύ τις δυνατότητες μιας τυπικής GPU 8GB. Τέταρτον, η διεύρυνση των target modules οδήγησε σε χαμηλότερο eval loss, αλλά με τίμημα αυξημένο χρόνο εκπαίδευσης. Τα ευρήματα αυτά δείχνουν ότι η αύξηση της διαθέσιμης μνήμης δεν αναιρεί τα trade-offs, αλλά επιτρέπει τη διερεύνηση πιο απαιτητικών ρυθμίσεων.

4.13 Έλεγχος γενίκευσης (dataset_v1 → dataset_wiki_50k)

Για τον έλεγχο γενίκευσης σε δεδομένα εκτός πεδίου, εκτελέστηκαν επιπλέον πειράματα σε διαφορετικό corpus ελληνικών (dataset_wiki_50k) μετά το fine-tuning στο dataset_v1. Στον Πίνακα 4.6 παρουσιάζονται τόσο τα μετρικά εντός πεδίου (dataset_v1) όσο και τα μετρικά εκτός πεδίου (dataset_wiki_50k). Στα αποτελέσματα του TinyLlama (r=16) η QLoRA εμφανίζει ελαφρώς χαμηλότερο eval_loss και perplexity εκτός πεδίου σε σχέση με τη LoRA FP16, διατηρώντας παρόμοια απόδοση εντός πεδίου.

Env	Model	Method	r	L	in loss	in PPL	in acc	time (s)	VRAM	gen ds	gen loss	gen PPL	run_id
Colab L4	Gemma-2-2B	QLoRA	16	128	0.090	1.094	0.969	399.832	6847	dataset_wiki_50k	N/A	N/A	gen_l4_gemma_v1_to_wiki_qlora_r16
Colab L4	TinyLlama-1.1B	LoRA FP16	16	256	0.029	1.029	0.987	149.555	3031	dataset_wiki_50k	2.111	8.260	gen_l4_tiny_v1_to_wiki_lora_r16
Colab L4	TinyLlama-1.1B	QLoRA	16	256	0.029	1.029	0.987	286.202	1909	dataset_wiki_50k	2.065	7.882	gen_l4_tiny_v1_to_wiki_qlora_r16

Πίνακας 4.6: Πειράματα γενίκευσης (L4) – in-domain και out-of-domain metrics (wiki_50k).

4.14 Αντίστροφος έλεγχος γενίκευσης (dataset_wiki_50k → dataset_v1)

Για να εξεταστεί η συμπεριφορά γενίκευσης υπό αντιστροφή domain, εκτελέστηκαν πειράματα όπου το μοντέλο εκπαιδεύεται στο dataset_wiki_50k και αξιολογείται ως προς τη γενίκευση στο dataset_v1. Στόχος εδώ είναι να φανεί αν η σχέση μνήμης και χρόνου ανάμεσα σε LoRA FP16 και QLoRA συνοδεύεται ή όχι από ουσιαστική διαφορά στη μεταφορά σε δεύτερο σύνολο δεδομένων.

Τα αποτελέσματα συνοψίζονται στον Πίνακα 4.7.

Method	PPL (wiki)	Gen PPL (v1)	time (h)	VRAM (GiB)	run_id
lora_fp16	3.027	2.916	2.18	2.96	gen_l4_tiny_wiki_to_v1_lora_r16_rev2
qlora	3.059	2.907	4.15	1.86	gen_l4_tiny_wiki_to_v1_qlora_r16_rev2

Πίνακας 4.7: Αντίστροφος έλεγχος γενίκευσης (dataset_wiki_50k → dataset_v1) – αποτελέσματα (L4).

Στο συγκεκριμένο σενάριο η QLoRA μειώνει την peak VRAM, αλλά με αυξημένο χρόνο εκπαίδευσης, ενώ η γενίκευση στο dataset_v1 παραμένει πρακτικά ισοδύναμη με εκείνη της LoRA FP16.

4.15 Συνεχιζόμενη προεκπαίδευση τύπου DAPT σε ελληνικό corpus

Στο δεύτερο σκέλος των πειραμάτων δοκιμάστηκε ένα μικρό σενάριο συνεχιζόμενης προεκπαίδευσης τύπου DAPT (Domain-Adaptive PreTraining) σε ελληνικό corpus. Η εκτέλεση αυτή δεν ισοδυναμεί με πλήρες pretraining μεγάλης κλίμακας. Χρησιμοποιείται περισσότερο ως ελεγχόμενη δοκιμή: εξετάζει αν μια ενδιαμέση έκθεση σε ελληνικό υλικό μπορεί να επηρεάσει τη γενίκευση πριν ακολουθήσει νέο fine-tuning.

Για το στάδιο αυτό χρησιμοποιήθηκε το dataset_pretrain_el, το οποίο περιείχε ελληνικό υλικό από τη Wikipedia. Η εκτέλεση έγινε με QLoRA σε Colab L4, κυρίως επειδή οι διαθέσιμοι πόροι δεν επέτρεπαν πιο βαριές ρυθμίσεις. Οι βασικές υπερπαραμέτροι ήταν max_length=128, batch=2 και grad_accum=4. Το DAPT στάδιο εκτελέστηκε για 3500 steps με max_train_samples=60000 και στη συνέχεια ακολούθησε fine-tuning στο dataset_v1 για 1500 steps, ξεκινώντας από τον adapter που είχε παραχθεί στο πρώτο στάδιο.

Η αποτίμηση του DAPT δεν στηρίχθηκε σε μία μόνο ένδειξη. Αρχικά εξετάστηκε η επίδοση στο corpus προεκπαίδευσης, ώστε να ελεγχθεί αν το μοντέλο προσαρμόζεται πράγματι στο ελληνικό υλικό. Έπειτα εξετάστηκε η συμπεριφορά εκτός πεδίου στο dataset_wiki_50k και, όπου υπήρχαν διαθέσιμα αποτελέσματα, στο dataset_v1. Με αυτά τα βήματα, το πείραμα

δίνει μια πρώτη εικόνα για τη σχέση ανάμεσα στην ενδιάμεση προεκπαίδευση, τη γενίκευση και τη μεταγενέστερη εξειδίκευση μέσω fine-tuning.

Τα αποτελέσματα συνοψίζονται στον Πίνακα 4.8.

Stage	Train dataset	Len	Steps	Eval PPL	Gen PPL (wiki)	Gen PPL (v1)	time (h)	run_id	VRAM (GiB)
DAPT (continued pretraining-like)	dataset_pretrain_el	128	3500	3.967	3.842	2.717	1.26	dapt_l4_tiny_wikiel_qlora_r16_len128_s3500_b2	1.43
Stage-2 (DAPT→fine-tune στο dataset_v1)	dataset_v1	128	1500	1.023	9.069	—	0.55	dapt2_l4_tiny_wikiel_to_v1_qlora_r16_len128_b2	1.44
Baseline (μόνο dataset_v1)	dataset_v1	256	118	1.029	7.882	—	0.08	gen_l4_tiny_v1_to_wiki_qlora_r16	1.86

Πίνακας 4.8: Αποτελέσματα DAPT και Stage-2 (DAPT→fine-tuning) σε ελληνικό corpus.

Στα αποτελέσματα, το DAPT στάδιο πέτυχε χαμηλότερο perplexity στο dataset_wiki_50k σε σύγκριση με το αντίστοιχο πείραμα όπου είχε προηγηθεί μόνο fine-tuning στο dataset_v1. Η διαφορά αυτή δείχνει ότι η πρόσθετη έκθεση σε ευρύτερο ελληνικό υλικό μπορεί να βοηθήσει τη συμπεριφορά εκτός πεδίου. Μετά το δεύτερο στάδιο, δηλαδή το DAPT→fine-tuning στο dataset_v1, το perplexity στο dataset_wiki_50k αυξήθηκε ξανά. Η μεταβολή αυτή είναι αναμενόμενη: το μοντέλο μετακινήθηκε προς το μικρότερο και πιο ειδικό dataset, χάνοντας μέρος της προηγούμενης γενίκευσης.

Τα αποτελέσματα αυτά πρέπει να διαβαστούν με προσοχή. Το DAPT run έγινε με max_length=128, επειδή η διαθέσιμη μνήμη δεν επέτρεπε μεγαλύτερο context, και περιλάμβανε σχετικά λίγα βήματα. Για continued pretraining, ένα τέτοιο παράθυρο συμφραζομένων είναι μικρό, καθώς κόβει μεγαλύτερες εξαρτήσεις μέσα στο κείμενο. Σε επόμενη δοκιμή θα είχε νόημα να χρησιμοποιηθεί QLoRA με paged optimizers και μήκος 1024 ή 2048 tokens, ακόμη και με batch size 1, ώστε να ελεγχθεί αν το μεγαλύτερο context βοηθά την προσαρμογή στο ελληνικό corpus χωρίς να ξεπεραστούν τα διαθέσιμα όρια VRAM. Γι' αυτό το DAPT σκέλος αντιμετωπίζεται ως αρχική ένδειξη και όχι ως πλήρης μελέτη continued pretraining.

Πέρα από την εκτέλεση στο Colab L4, έγινε και μια πιο χαμηλού κόστους δοκιμή στο τοπικό περιβάλλον με RTX 3070. Η τοπική εκτέλεση είχε κυρίως χαρακτήρα ελέγχου εφικτότητας. Δείχνει ότι η ίδια λογική μπορεί να αναπαραχθεί και εκτός Colab, αλλά δεν αναιρεί τον περιορισμένο χαρακτήρα του πειράματος. Το DAPT χρησιμοποιείται εδώ ως συμπληρωματική διερεύνηση της προσαρμογής σε ελληνικό corpus και όχι ως ανεξάρτητο benchmark προεκπαίδευσης.

4.16 Διερεύνηση κλιμάκωσης σε A100 80GB (VRAM)

Για να εξεταστεί πιο καθαρά το ζήτημα της VRAM, εκτελέστηκε ένα πρόσθετο πείραμα σε A100 80GB με LoRA FP16, μεγαλύτερο context (max_length=1024) και μεγάλο batch. Μια τέτοια ρύθμιση δεν είναι ρεαλιστική για GPUs περίπου 8GB· χρησιμοποιείται όμως ως σημείο αναφοράς για το πώς κλιμακώνονται οι απαιτήσεις όταν αυξάνεται ταυτόχρονα το μήκος ακολουθίας και το batch.

Η ρύθμιση και τα αποτελέσματα συνοψίζονται στον Πίνακα 4.9.

run_id	Method	Len	Batch	Steps	VRAM (GiB)	time (h)	Eval PPL	Gen PPL (v1)	Gen PPL (wiki)
a100_80gb_vram50plus_lora_fp16_tiny_len1024_b64_s100	lora_fp16	1024	64	100	31.66	0.12	4.828	2.680	3.201

Πίνακας 4.9: Πείραμα κλιμάκωσης σε A100 80GB – VRAM scaling (Len=1024, Batch=64).

Στο run a100_80gb_vram50plus_lora_fp16_tiny_len1024_b64_s100 καταγράφηκε peak VRAM περίπου 31.66 GiB και χρόνος εκτέλεσης περίπου 0.12 h για 100 steps. Το αποτέλεσμα είναι ενδεικτικό: ακόμη και με μοντέλο 1.1B, ο συνδυασμός FP16, μεγάλου context και μεγάλου batch αυξάνει πολύ γρήγορα τις απαιτήσεις μνήμης, ειδικά όταν η ρύθμιση πλησιάζει περισσότερο σε λογική προεκπαίδευσης παρά σε ελαφρύ fine-tuning.

Μετά τα βασικά πειράματα προστέθηκε ένα μικρό σύνολο εκτελέσεων με Qwen2.5-1.5B-Instruct, Phi-3.5 Mini, Meltani 7B και Krikri 8B. Η προσθήκη αυτή δεν επιχειρεί συνολική κατάταξη των μοντέλων. Χρησιμοποιείται κυρίως για να ελεγχθεί αν οι ίδιες αντισταθμίσεις ανάμεσα σε LoRA και QLoRA εμφανίζονται και όταν αλλάζουν η αρχιτεκτονική, το μέγεθος του μοντέλου και το διαθέσιμο περιθώριο μνήμης.

Οι εκτελέσεις του Πίνακα 4.10 έγιναν σε Colab με NVIDIA A100-SXM4-40GB, με dataset_v1 και rank r=16. Στο Qwen2.5-1.5B χρησιμοποιήθηκαν max_length=256, batch=1

και `grad_accum=8`. Για τα Phi-3.5 Mini, Meltemi 7B και Krikri 8B επιλέχθηκε πιο συντηρητική ρύθμιση, με `max_length=128`, `batch=1` και `grad_accum=16`. Στα Qwen και Phi δοκιμάστηκαν και οι δύο μέθοδοι, LoRA FP16 και QLoRA. Στα Meltemi και Krikri η αξιολόγηση περιορίστηκε στην QLoRA, καθώς μια αντίστοιχη LoRA FP16 εκτέλεση θα απαιτούσε πολύ μεγαλύτερο περιθώριο μνήμης.

Μοντέλο	Μέθοδος	PPL	Eval loss	Token acc.	Χρόνος (s)	Peak VRAM (MiB)
Qwen2.5-1.5B	QLoRA	1.0417	0.0409	0.9921	357.0	5737
Qwen2.5-1.5B	LoRA FP16	1.0141	0.0140	0.9942	197.6	7152
Phi-3.5 Mini	QLoRA	1.0243	0.0240	0.9861	325.9	3065
Phi-3.5 Mini	LoRA FP16	1.0236	0.0234	0.9854	211.7	7875
Meltemi 7B	QLoRA	1.6140	0.4787	0.9203	424.7	6291
Krikri 8B	QLoRA	2.1342	0.7581	0.8931	418.0	10232

Πίνακας 4.10: Συμπληρωματικά αποτελέσματα πρόσθετων μοντέλων σε Colab A100.

Στο Qwen2.5-1.5B η LoRA FP16 πέτυχε χαμηλότερη perplexity και μικρότερο χρόνο εκπαίδευσης από την QLoRA, με κόστος την υψηλότερη χρήση VRAM. Στο Phi-3.5 Mini η ποιοτική διαφορά ήταν πρακτικά πολύ μικρή, ενώ η QLoRA διατήρησε σαφές πλεονέκτημα στη μνήμη. Τα Meltemi 7B και Krikri 8B ολοκληρώθηκαν σε A100 με QLoRA, αλλά στο συγκεκριμένο setup εμφάνισαν υψηλότερη perplexity και χαμηλότερη token-level accuracy από τα Qwen και Phi.

Στο τοπικό σύστημα οι περιορισμοί έγιναν πιο εμφανείς. Στην RTX 3070 8GB ολοκληρώθηκαν τα δύο Qwen2.5-1.5B runs και το Phi-3.5 Mini με QLoRA. Αντίθετα, το Phi-3.5 Mini με LoRA FP16 δεν ολοκληρώθηκε, λόγω offloading/meta-device σφάλματος, ενώ το Meltemi 7B με QLoRA σταμάτησε με Out-Of-Memory. Επομένως, η QLoRA μετακινεί αρκετά τα όρια εφικτότητας, αλλά δεν αρκεί πάντα για μοντέλα 7B-8B όταν η διαθέσιμη VRAM περιορίζεται στα 8GB.

Επειδή οι αριθμητικές μετρικές δεν δείχνουν από μόνες τους αν μια απάντηση είναι χρήσιμη, έγινε και ένας μικρός χειροκίνητος έλεγχος παραγόμενων απαντήσεων. Χρησιμοποιήθηκαν τέσσερις ερωτήσεις εντός πεδίου, γύρω από fine-tuning, LoRA, QLoRA και περιορισμούς μνήμης GPU. Οι απαντήσεις βαθμολογήθηκαν από 1 έως 5 ως προς την ορθότητα, την πληρότητα, τη σαφήνεια, την ποιότητα των ελληνικών και τη συνολική εντύπωση. Η μέση συνολική βαθμολογία ήταν 2,4/5 για το QLoRA r16 και 2,0/5 για το LoRA FP16 r16.

Η μικρή αυτή αξιολόγηση έδειξε ότι οι adapters απορρόφησαν μέρος του λεξιλογίου και της θεματολογίας, χωρίς όμως να παράγουν σταθερά ώριμες απαντήσεις. Σε ορισμένα παραδείγματα εμφανίστηκαν επαναλήψεις, ασαφής χρήση όρων και αδύναμη διάκριση

ανάμεσα σε LoRA και QLoRA. Επομένως, το loss και το perplexity πρέπει να διαβάζονται ως μετρήσεις πιθανοτικής προσαρμογής στο evaluation set και όχι ως εγγύηση ότι οι παραγόμενες απαντήσεις θα είναι πάντα σωστές ή χρήσιμες για τελικό χρήστη.

5. Συζήτηση

5.1 LoRA έναντι QLoRA: αντιστάθμιση ποιότητας, μνήμης και χρόνου

Στην RTX 3070 φαίνεται καθαρά ότι η QLoRA μειώνει αισθητά τη χρήση VRAM, ιδιαίτερα στο TinyLlama, και έτσι κάνει το fine-tuning πιο εφικτό σε περιβάλλοντα με περιορισμένη μνήμη. Αντίστοιχη εικόνα προκύπτει και στο Colab L4. Το όφελος αυτό, όμως, δεν έρχεται χωρίς κόστος: στις περισσότερες περιπτώσεις η QLoRA χρειάζεται περισσότερο χρόνο εκπαίδευσης, ενώ στο Gemma-2-2B της τοπικής εκτέλεσης συνδέθηκε και με αισθητή επιδείνωση στην perplexity. Συνεπώς, η σύγκριση LoRA–QLoRA δεν καταλήγει σε μία καθολικά καλύτερη λύση· εξαρτάται από το αν ο βασικός περιορισμός είναι η μνήμη, ο χρόνος ή η ποιότητα.

5.2 Επίδραση του rank r

Η αύξηση του rank r από 8 σε 16 και 32 συνοδεύεται, στις περισσότερες περιπτώσεις, από μικρές βελτιώσεις στην απόδοση (χαμηλότερο perplexity). Μετά το $r=16$, όμως, τα οφέλη περιορίζονται. Για τον λόγο αυτό, το $r \approx 16$ λειτουργεί ως μια λογική ισορροπία ανάμεσα σε απόδοση και κόστος.

5.3 Επίδραση υπολογιστικού περιβάλλοντος

Τα αποτελέσματα στο A100 (Πίνακας 4.4) δείχνουν ότι η σχέση LoRA–QLoRA ως προς μνήμη και χρόνο παραμένει ουσιαστικά η ίδια. Οι εκτελέσεις αναπαραγωγής δείχνουν επίσης ότι οι χρόνοι L4 και A100 είναι πολύ κοντά όταν οι υπερπαράμετροι κρατιούνται ίδιες. Αντίθετα, η RTX 3070 αποδείχθηκε αρκετά ανταγωνιστική στο TinyLlama, κάτι που έχει πρακτικό ενδιαφέρον για μικρότερα μοντέλα και πιο περιορισμένα πειράματα.

Στο Colab L4 επανέρχεται η ίδια γενική εικόνα: η QLoRA μειώνει την peak VRAM σε σχέση με τη LoRA FP16, αλλά συνήθως αυξάνει τον χρόνο εκπαίδευσης. Για $r=16$, στο TinyLlama-1.1B η QLoRA μείωσε την peak VRAM κατά περίπου 37.2% και αύξησε τον χρόνο εκπαίδευσης κατά περίπου 88.9%. Στο Gemma-2-2B η μείωση της peak VRAM ήταν περίπου 20.5%, ενώ η αύξηση του χρόνου ήταν περίπου 59.3%. Τα στοιχεία αυτά δεν αλλάζουν το βασικό συμπέρασμα, αλλά το κάνουν πιο συγκεκριμένο.

Τα συμπληρωματικά πειράματα στο A100 με Qwen2.5-1.5B, Phi-3.5 Mini, Meltani 7B και Krikri 8B διευρύνουν την ίδια εικόνα. Στο Qwen η LoRA FP16 ήταν ταχύτερη και έδωσε

καλύτερες μετρικές, ενώ η QLoRA κράτησε το αναμενόμενο πλεονέκτημα μνήμης. Στο Phi οι δύο μέθοδοι είχαν σχεδόν ίδια perplexity, με σαφώς χαμηλότερη VRAM για την QLoRA. Τα Meltemi και Krikri, παρότι είναι ελληνικά ή ελληνόφωνα και μεγαλύτερα, δεν έδωσαν καλύτερες μετρικές στο συγκεκριμένο setup. Το αποτέλεσμα δείχνει ότι το μέγεθος και ο γλωσσικός προσανατολισμός δεν αρκούν από μόνα τους· σημαντικό ρόλο παίζουν επίσης η ρύθμιση του fine-tuning, τα δεδομένα και ο tokenizer.

5.4 Πλαίσιο επιλογής τεχνικής ανά διαθέσιμους πόρους

Με βάση τα πειραματικά αποτελέσματα της εργασίας, η επιλογή ανάμεσα σε LoRA FP16 και QLoRA δεν μπορεί να θεωρηθεί ανεξάρτητη από το διαθέσιμο υπολογιστικό περιβάλλον. Οι δύο τεχνικές εξυπηρετούν διαφορετικές πρακτικές ανάγκες και η καταλληλότητά τους εξαρτάται από τον βασικό περιορισμό κάθε πειραματικού σεναρίου: τη διαθέσιμη VRAM, τον χρόνο εκπαίδευσης, το μέγεθος του μοντέλου, το μήκος ακολουθίας και τον στόχο της αξιολόγησης.

Σε περιβάλλοντα με περιορισμένη μνήμη GPU, όπως κάρτες της τάξης των 8–10 GB VRAM, η QLoRA αποτελεί πιο ασφαλή επιλογή, επειδή η 4-bit ποσοτικοποίηση της βάσης του μοντέλου μειώνει σημαντικά το αποτύπωμα μνήμης. Αυτό είναι ιδιαίτερα χρήσιμο όταν ο στόχος είναι να εκτελεστεί fine-tuning χωρίς σφάλματα Out-Of-Memory ή όταν χρειάζεται να διερευνηθούν πολλοί συνδυασμοί υπερπαραμέτρων με περιορισμένο εξοπλισμό. Στα πειράματα της εργασίας, η QLoRA εμφάνισε σταθερό πλεονέκτημα ως προς τη μέγιστη χρήση VRAM, γεγονός που την καθιστά πρακτική επιλογή για αρχική διερεύνηση, πειράματα εφικτότητας και εργασίες σε περιβάλλοντα χαμηλού υπολογιστικού προϋπολογισμού.

Αντίθετα, όταν υπάρχει επαρκές περιθώριο μνήμης και ο βασικός περιορισμός είναι ο χρόνος εκπαίδευσης, η LoRA FP16 μπορεί να είναι προτιμότερη. Στα περισσότερα πειράματα, η LoRA FP16 ήταν ταχύτερη από την QLoRA, επειδή αποφεύγει το πρόσθετο υπολογιστικό κόστος που σχετίζεται με την ποσοτικοποίηση και την αποποσοτικοποίηση κατά την εκτέλεση. Επομένως, σε περιβάλλοντα με μεγαλύτερη VRAM ή όταν το μοντέλο χωρά άνετα σε FP16, η LoRA FP16 προσφέρει συχνά καλύτερη ισορροπία μεταξύ απλότητας, ταχύτητας και σταθερότητας.

Η επιλογή επηρεάζεται επίσης από το μήκος ακολουθίας. Όσο αυξάνεται το max_length, αυξάνονται σημαντικά οι απαιτήσεις μνήμης, κυρίως λόγω της πολυπλοκότητας του self-

attention και της ανάγκης αποθήκευσης ενδιάμεσων ενεργοποιήσεων κατά την εκπαίδευση. Για μικρότερα μήκη, όπως 128 ή 256 tokens, είναι ευκολότερο να συγκριθούν LoRA FP16 και QLoRA ακόμη και σε πιο περιορισμένα περιβάλλοντα. Για μεγαλύτερα μήκη, όπως 512 ή 1024 tokens, απαιτείται μεγαλύτερο περιθώριο VRAM ή πιο προσεκτική επιλογή ρυθμίσεων, όπως QLoRA, μικρότερο batch size, gradient accumulation και gradient checkpointing. Τα πειράματα στο A100 δείχνουν ότι τέτοιες ρυθμίσεις είναι τεχνικά εφικτές, αλλά δεν αντιπροσωπεύουν απαραίτητα συνθήκες που μπορεί να αναπαραγάγει ένας χρήστης με καταναλωτική GPU.

Συνολικά, η QLoRA είναι καταλληλότερη όταν η μνήμη είναι ο κύριος περιοριστικός παράγοντας, ενώ η LoRA FP16 είναι συχνά προτιμότερη όταν υπάρχει αρκετή VRAM και ζητείται μικρότερος χρόνος εκπαίδευσης. Για πειράματα σε ελληνικά δεδομένα και περιβάλλοντα περιορισμένων πόρων, μια πρακτική στρατηγική είναι να ξεκινά η διερεύνηση με QLoRA, ώστε να εντοπίζονται ρεαλιστικές ρυθμίσεις χωρίς υψηλές απαιτήσεις μνήμης, και στη συνέχεια, εφόσον υπάρχουν διαθέσιμοι πόροι, να επαναλαμβάνονται τα πιο σημαντικά πειράματα με LoRA FP16 για ταχύτερη και πιο άμεση σύγκριση. Με αυτόν τον τρόπο, η επιλογή τεχνικής δεν αντιμετωπίζεται ως απόλυτη προτίμηση υπέρ μίας μεθόδου, αλλά ως απόφαση σχεδιασμού που εξαρτάται από το διαθέσιμο υλικό, τον χρόνο, το μέγεθος του μοντέλου και τον σκοπό του πειράματος.

5.5 Περιορισμοί και απειλές εγκυρότητας

Τα αποτελέσματα της παρούσας εργασίας πρέπει να ερμηνευθούν μέσα στο πλαίσιο των πειραματικών περιορισμών υπό τους οποίους πραγματοποιήθηκε η μελέτη. Ο βασικός στόχος δεν ήταν η εκτέλεση ενός πλήρους benchmark μεγάλης κλίμακας, αλλά η διερεύνηση της πρακτικής συμπεριφοράς τεχνικών PEFT σε περιβάλλοντα περιορισμένων και ετερογενών υπολογιστικών πόρων. Για τον λόγο αυτό, ορισμένες επιλογές σχεδιασμού, όπως ο μικρός αριθμός εποχών και η χρήση σχετικά περιορισμένου αρχικού συνόλου δεδομένων, αποτελούν ταυτόχρονα μέρος του πειραματικού πλαισίου και πηγή περιορισμών.

Πρώτος περιορισμός αφορά τον αριθμό εποχών εκπαίδευσης και το μέγεθος του dataset_v1. Τα βασικά πειράματα εκτελέστηκαν για μία εποχή, ενώ το dataset_v1 είναι μικρό σε σχέση με τα σύνολα δεδομένων που χρησιμοποιούνται συνήθως για εκτεταμένη προσαρμογή μεγάλων γλωσσικών μοντέλων. Η επιλογή αυτή επέτρεψε γρήγορες και επαναλήψιμες

συγκρίσεις σε διαφορετικές μεθόδους, ranks και υπολογιστικά περιβάλλοντα, αλλά είναι πιθανό να υποεκτιμά διαφορές ποιότητας που θα εμφανίζονταν μετά από περισσότερες εποχές ή μεγαλύτερα δεδομένα. Ειδικά οι διαφορές μεταξύ LoRA FP16 και QLoRA ενδέχεται να μεταβάλλονται όταν τα μοντέλα έχουν περισσότερο χρόνο να συγκλίνουν ή όταν εκτίθενται σε μεγαλύτερη ποικιλία γλωσσικών παραδειγμάτων.

Δεύτερος περιορισμός αφορά τις μετρικές αξιολόγησης. Το perplexity και το eval loss είναι κατάλληλες μετρικές για τη μέτρηση της ικανότητας πρόβλεψης επόμενου token, ενώ το token-level accuracy χρησιμοποιήθηκε ως συμπληρωματική ένδειξη ακρίβειας σε επίπεδο token. Ωστόσο, το token-level accuracy δεν ισοδυναμεί με αξιολόγηση σε πραγματικές downstream εργασίες, όπως question answering, ταξινόμηση κειμένου ή παραγωγή απαντήσεων με σημασιολογικά κριτήρια ποιότητας. Επομένως, τα αποτελέσματα δείχνουν κυρίως τη συμπεριφορά των μεθόδων στο πλαίσιο γλωσσικής μοντελοποίησης και όχι πλήρη αξιολόγηση της χρησιμότητας των προσαρμοσμένων μοντέλων σε εφαρμογές τελικού χρήστη.

Η χειροκίνητη αξιολόγηση, παρότι μικρής κλίμακας, δείχνει τον ίδιο περιορισμό από πιο πρακτική πλευρά. Ακόμη και σε ερωτήσεις εντός πεδίου, οι συνολικές βαθμολογίες 2,4/5 για QLoRA r16 και 2,0/5 για LoRA FP16 r16 δείχνουν ότι μια καλή ποσοτική εικόνα δεν μετατρέπεται αυτόματα σε καλή απάντηση φυσικής γλώσσας. Για ασφαλέστερα συμπεράσματα θα χρειαζόνταν περισσότερες ερωτήσεις, περισσότεροι αξιολογητές και πιο αυστηρά ορισμένο πρωτόκολλο ανθρώπινης αξιολόγησης.

Τρίτος περιορισμός σχετίζεται με τη συγκρισιμότητα των πειραμάτων μεταξύ διαφορετικών περιβαλλόντων. Στο Gemma-2-2B, η αρχική χρήση μεγαλύτερου μήκους ακολουθίας οδήγησε σε περιορισμούς μνήμης, με αποτέλεσμα σε ορισμένες εκτελέσεις να χρησιμοποιηθεί max_length=128 για αποφυγή σφαλμάτων OOM. Η επιλογή αυτή ήταν αναγκαία ώστε να ολοκληρωθούν σταθερά τα πειράματα στο Colab L4, αλλά σημαίνει ότι οι χρόνοι εκπαίδευσης και η χρήση VRAM δεν είναι πάντοτε άμεσα συγκρίσιμα με πειράματα διαφορετικού max_length. Για τον λόγο αυτό, όπου ήταν εφικτό, πραγματοποιήθηκαν επιπλέον εκτελέσεις με κοινές υπερπαραμέτρους, ώστε η σύγκριση να βασίζεται σε πιο ελεγχόμενες συνθήκες.

Ένας ακόμη παράγοντας που μπορεί να επηρεάσει τη χρονική σύγκριση είναι η επιλογή attention kernel. Στο Gemma-2-2B χρησιμοποιήθηκε σε ορισμένες περιπτώσεις eager

attention για λόγους συμβατότητας και σταθερότητας, ιδιαίτερα σε συνδυασμό με ποσοτικοποίηση 4-bit και LoRA adapters. Η επιλογή αυτή βοήθησε στην αποφυγή ασυμβατοτήτων τύπων δεδομένων στη διαδρομή SDPA, αλλά ενδέχεται να έχει κόστος σε χρόνο εκτέλεσης σε σχέση με πιο βελτιστοποιημένα attention kernels. Συνεπώς, οι μετρήσεις χρόνου πρέπει να διαβάζονται όχι μόνο ως χαρακτηριστικό της μεθόδου LoRA ή QLoRA, αλλά και ως αποτέλεσμα του συγκεκριμένου λογισμικού και kernel configuration.

Περιορισμένο εύρος έχουν και τα DAPT πειράματα. Έγιναν με μικρό υπολογιστικό προϋπολογισμό, max_length=128 και σχετικά λίγα βήματα, ώστε να παραμείνουν εφικτά σε Colab L4. Το μικρό context length είναι ουσιαστικός περιορισμός, επειδή τα σύγχρονα γλωσσικά μοντέλα συχνά αξιοποιούν μεγαλύτερα παράθυρα συμφραζομένων. Μια επόμενη δοκιμή με QLoRA paged optimizers, μήκος 1024 ή 2048 και batch size 1 θα μπορούσε να δείξει αν το μεγαλύτερο context βελτιώνει την προσαρμογή στο ελληνικό corpus χωρίς υπέρβαση των ορίων μνήμης. Επομένως, τα αποτελέσματα του DAPT πρέπει να θεωρηθούν αρχική ένδειξη και όχι πλήρης μελέτη continued pretraining.

Ανάλογη επιφύλαξη χρειάζεται και για τα πρόσθετα πειράματα με Qwen, Phi, Meltani και Krikri. Πολλές εκτελέσεις έγιναν στο ίδιο A100, κάτι που βοηθά τη σύγκριση εφικτότητας, όμως οι ρυθμίσεις παρέμειναν συντηρητικές: μία εποχή, μικρό evaluation subset και διαφορετικό max_length ανάλογα με το μοντέλο. Τα αποτελέσματα, επομένως, δείχνουν τη συμπεριφορά των LoRA/QLoRA σε διαφορετικές αρχιτεκτονικές και όχι ποιο μοντέλο είναι γενικά καλύτερο για την ελληνική γλώσσα.

Τέλος, πρέπει να ληφθεί υπόψη ο μη ντετερμινισμός που χαρακτηρίζει την εκπαίδευση σε GPU. Παρότι χρησιμοποιήθηκε σταθερό seed και τυποποιημένη δομή πειραμάτων, μικρές αποκλίσεις μπορούν να προκύψουν από GPU kernels, εκδόσεις βιβλιοθηκών, επιλογές attention implementation και λεπτομέρειες του περιβάλλοντος εκτέλεσης. Επομένως, τα συμπεράσματα της εργασίας πρέπει να ερμηνεύονται ως τεκμηριωμένες ενδείξεις για τις πρακτικές αντισταθμίσεις μνήμης, χρόνου και ποιότητας ανάμεσα σε LoRA FP16 και QLoRA, και όχι ως απόλυτη κατάταξη των μεθόδων σε κάθε πιθανή ρύθμιση.

6. Συμπεράσματα

Η εργασία έδειξε ότι οι τεχνικές PEFT μπορούν να κάνουν το fine-tuning LLMs σε ελληνικά δεδομένα πρακτικά εφικτό ακόμη και με περιορισμένους πόρους. Στο TinyLlama-1.1B η QLoRA έδωσε καθαρό όφελος μνήμης με μικρές ποιοτικές διαφορές, επομένως αποτελεί ρεαλιστική επιλογή για GPUs χαμηλής VRAM. Στο Gemma-2-2B η εικόνα ήταν πιο σύνθετη: η LoRA FP16 είχε καλύτερη perplexity στην τοπική εκτέλεση, ενώ η QLoRA κράτησε το πλεονέκτημα της χαμηλότερης μνήμης, αλλά με μεγαλύτερο χρόνο. Το DAPT σκέλος έδειξε ότι μια ενδιάμεση έκθεση σε ελληνικό corpus μπορεί να βελτιώσει προσωρινά την εκτός πεδίου συμπεριφορά, πριν το επόμενο fine-tuning μετακινήσει ξανά το μοντέλο προς το μικρότερο και ειδικότερο dataset.

Οι πρόσθετες εκτελέσεις με Qwen2.5-1.5B-Instruct, Phi-3.5 Mini, Meltemi 7B και Krikri 8B έδωσαν ένα δεύτερο επίπεδο ελέγχου των ίδιων συμπερασμάτων. Το βασικό εύρημα παρέμεινε σταθερό: η QLoRA είναι ιδιαίτερα χρήσιμη όταν η μνήμη είναι ο κύριος περιορισμός, ενώ η LoRA FP16 μπορεί να είναι ταχύτερη ή ελαφρώς καλύτερη ποιοτικά όταν υπάρχει αρκετή VRAM. Τα Meltemi και Krikri έδειξαν επίσης ότι ο ελληνικός προσανατολισμός ή το μεγαλύτερο μέγεθος δεν εγγυώνται χαμηλότερη perplexity στο συγκεκριμένο setup. Για συνέχεια θα είχε νόημα μεγαλύτερο ελληνικό corpus, περισσότερες εποχές, μεγαλύτερα context windows και πιο συστηματική ανθρώπινη αξιολόγηση.

Βιβλιογραφία

Ελληνική βιβλιογραφία

Ελληνικό Ανοικτό Πανεπιστήμιο. (2018). Προδιαγραφές συγγραφής πτυχιακών εργασιών και διπλωματικών εργασιών. Πάτρα: Ελληνικό Ανοικτό Πανεπιστήμιο.

Ελληνικό Ανοικτό Πανεπιστήμιο. (2020). Ειδικός Κανονισμός Εκπόνησης Μεταπτυχιακών Διπλωματικών Εργασιών (ΜΔΕ) για το ΠΣ «Μεταπτυχιακή Εξειδίκευση στα Πληροφοριακά Συστήματα» (ΠΛΣ). Πάτρα: Ελληνικό Ανοικτό Πανεπιστήμιο.

Evdaïmon, I., Abdine, H., Xypolopoulos, C., Outsios, S., Vazirgiannis, M., & Stamou, G. (2024). GreekBART: The first pretrained Greek sequence-to-sequence model. In LREC-COLING 2024 (pp. 7949–7962).

Gavrilidou, M. (2002). The Hellenic National Corpus on-line. *Revue belge de philologie et d'histoire*, 80(3).

Koutsikakis, J., Chalkidis, I., Malakasiotis, P., & Androutsopoulos, I. (2020). GREEK-BERT: The Greeks visiting Sesame Street. In 11th Hellenic Conference on Artificial Intelligence (SETN 2020). arXiv:2008.12014.

Roussis, D., Voukoutis, L., Paraskevopoulos, G., Sofianopoulos, S., Prokopidis, P., Papavasileiou, V., Katsamanis, A., Piperidis, S., & Katsouros, V. (2025). Krikri: Advancing open large language models for Greek. Findings of the Association for Computational Linguistics: EMNLP 2025.

Saketos, V., Pantazi, D.-A., & Koubarakis, M. (2024). The large language model GreekLegalRoBERTa. arXiv:2410.12852.

Voukoutis, L., Roussis, D., Paraskevopoulos, G., Sofianopoulos, S., Prokopidis, P., Papavasileiou, V., Katsamanis, A., Piperidis, S., & Katsouros, V. (2024). Meltemi: The first open large language model for Greek. arXiv:2407.20743.

Ξενόγλωσση βιβλιογραφία (APA)

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., ... Amodei, D. (2020). Language

models are few-shot learners. *Advances in Neural Information Processing Systems*, 33, 1877–1901.

Chung, H. W., Hou, L., Longpre, S., Zoph, B., Tay, Y., Fedus, W., Li, E., Wang, X., Dehghani, M., Brahma, S., Webson, A., Gu, S., Dai, Z., Suzgun, M., Chen, X., Chowdhery, A., Narang, S., Mishra, G., Yu, A., ... Wei, J. (2022). Scaling instruction-finetuned language models. *arXiv:2210.11416*.

Conover, J., Hayes, M., Mathur, A., Meng, J., & others. (2023). Databricks Dolly: An instruction-following LLaMA model (Technical report / blog).

Dettmers, T., Pagnoni, A., Holtzman, A., & Zettlemoyer, L. (2023). QLoRA: Efficient finetuning of quantized LLMs. *Advances in Neural Information Processing Systems*. *arXiv:2305.14314*.

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. *Proceedings of NAACL-HLT 2019*, 4171–4186.

Gemma Team. (2024). Gemma 2: Improving open language models at a practical size (Technical report). Google DeepMind.

Gururangan, S., Marasović, A., Swayamdipta, S., Lo, K., Beltagy, I., Downey, D., & Smith, N. A. (2020). Don't stop pretraining: Adapt language models to domains and tasks. *ACL 2020*. *arXiv:2004.10964*.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., & Chen, W. (2021). LoRA: Low-rank adaptation of large language models. *arXiv:2106.09685*.

Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., & Amodei, D. (2020). Scaling laws for neural language models. *arXiv:2001.08361*.

Lhoest, Q., Villanova del Moral, A., Jernite, Y., Thakur, A., von Platen, P., Patil, S., Chaudhary, S., Gupta, J., Sparks, E., Damania, P., & others. (2021). Datasets: A community library for natural language processing. In *Proceedings of EMNLP 2021: System Demonstrations*.

- Li, X. L., & Liang, P. (2021). Prefix-tuning: Optimizing continuous prompts for generation. *Proceedings of ACL-IJCNLP 2021*, 4582–4597.
- Liu, X., Ji, K., Fu, Y., Tam, W. L., Du, Z., Yang, Z., & Tang, J. (2022). P-Tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. *arXiv:2110.07602*.
- Loshchilov, I., & Hutter, F. (2019). Decoupled weight decay regularization. *Proceedings of ICLR 2019*.
- Microsoft. (2024). Phi-3.5-mini-instruct [Model card]. Hugging Face. <https://huggingface.co/microsoft/Phi-3.5-mini-instruct>
- Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C. L., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., & Lowe, R. (2022). Training language models to follow instructions with human feedback. *NeurIPS 2022*. *arXiv:2203.02155*.
- Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2019). Language models are unsupervised multitask learners. *OpenAI Technical Report*.
- Qwen Team. (2024). Qwen2.5 technical report. *arXiv:2412.15115*.
- Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., & Liu, P. J. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140), 1–67. *arXiv:1910.10683*.
- Sanh, V., Webson, A., Raffel, C., Bach, S. H., Sutawika, L., Alyafeai, Z., Chaffin, A., Stiegler, A., Le Scao, T., Raja, A., Dey, M., Bari, M. S., Xu, C., Thakker, U., Sharma, S. S., Szczechla, E., Kim, T., Chhablani, G., Nayak, N., ... Rush, A. M. (2022). Multitask prompted training enables zero-shot task generalization. *Proceedings of ICLR 2022*.
- Scao, T. L., Fan, A., Akiki, C., Pavlick, E., Ilić, S., Hesslow, D., Castagné, R., Luccioni, A. S., Yvon, F., Gallé, M., Tow, J., Rush, A. M., Biderman, S., Webson, A., Ammanamanchi, P. S., Wang, T., Sagot, B., Muennighoff, N., Moral, A. V., ... Wolf, T. (2022). BLOOM: A 176B-parameter open-access multilingual language model. *arXiv:2211.05100*.
- Shazeer, N. (2020). GLU variants improve transformer. *arXiv:2002.05202*.

- Su, J., Lu, Y., Pan, S., Wen, B., & Liu, Y. (2021). RoFormer: Enhanced transformer with rotary position embedding. arXiv:2104.09864.
- Taori, R., Gulrajani, I., Zhang, T., Dubois, Y., Li, X., Guestrin, C., Liang, P., & Hashimoto, T. B. (2023). Stanford Alpaca: An instruction-following LLaMA model (Technical report).
- Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., et al. (2023). Llama 2: Open foundation and fine-tuned chat models. arXiv:2307.09288.
- Tunstall, L., Beeching, E., Lambert, N., & others. (2023). Zephyr: Direct distillation of language model alignment (Technical report).
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. Advances in Neural Information Processing Systems.
- Wei, J., Bosma, M., Zhao, V. Y., Guu, K., Yu, A. W., Lester, B., Du, N., Dai, A. M., & Le, Q. V. (2022). Finetuned language models are zero-shot learners. Proceedings of ICLR 2022.
- Wikimedia Foundation. (2023). Wikipedia dumps: elwiki 20231101. Retrieved from dumps.wikimedia.org.
- Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., & Brew, J. (2020). Transformers: State-of-the-art natural language processing. In Proceedings of EMNLP 2020: System Demonstrations.
- Zhang, P., Zeng, G., Wang, T., & Lu, W. (2024). TinyLlama: An open-source small language model. arXiv:2401.02385.
- Zhang, S., Roller, S., Goyal, N., Artetxe, M., Chen, M., Chen, S., Dewan, C., Diab, M., Li, X., Lin, X. V., Mihaylov, T., Ott, M., Shleifer, S., Shuster, K., Simig, D., Koura, P. S., Sridhar, A., Wang, T., & Zettlemoyer, L. (2022). OPT: Open pre-trained transformer language models. arXiv:2205.01068.

Παράρτημα Α: Κώδικας εκτέλεσης πειραμάτων και ενοποίησης αποτελεσμάτων

Το παράρτημα περιγράφει συνοπτικά την πειραματική υλοποίηση των LoRA FP16 και QLoRA runs, χωρίς να παραθέτει ολόκληρο τον πηγαίο κώδικα. Ο πλήρης κώδικας διατηρείται στο συνοδευτικό υλικό, ώστε τα πειράματα να μπορούν να αναπαραχθούν. Στο κυρίως κείμενο μένουν μόνο τα στοιχεία που είναι απαραίτητα για να γίνει κατανοητή η λογική των scripts, η ροή εκτέλεσης και η δομή των παραγόμενων αρχείων.

A.1 Master runner (THESIS_MASTER_RUNS.py)

Το script THESIS_MASTER_RUNS.py χρησιμοποιήθηκε ως κεντρικός μηχανισμός για προκαθορισμένες ομάδες πειραμάτων. Ο ρόλος του ήταν να οργανώνει σειρές εκτελέσεων με κοινές ρυθμίσεις, καλώντας το βασικό script εκπαίδευσης για κάθε συνδυασμό μοντέλου, μεθόδου και LoRA rank. Έτσι μειώθηκε η πιθανότητα χειροκίνητων λαθών και διατηρήθηκε κοινή ονοματοδοσία στα πειράματα.

Οι βασικές ομάδες πειραμάτων περιλάμβαναν τις εκτελέσεις αναπαραγωγής σε Colab L4, τις αντίστοιχες εκτελέσεις σε Colab A100 και τις δοκιμές διερεύνησης δυνατοτήτων στο A100. Για κάθε run-set, το script παρήγαγε εντολές με κοινά πεδία, όπως μοντέλο, μέθοδο προσαρμογής, dataset, rank, max_length, batch size και gradient accumulation. Έτσι μειώθηκαν οι αποκλίσεις που θα μπορούσαν να προκύψουν από χειροκίνητη εισαγωγή παραμέτρων.

A.2 Runner script (run_peft_experiment.LOCAL.colab.py)

Το script run_peft_experiment.LOCAL.colab.py υλοποιούσε τον κύκλο εκπαίδευσης και αξιολόγησης. Ως είσοδο δεχόταν το μοντέλο, τη μέθοδο προσαρμογής, το dataset, το LoRA rank, τα target modules, το max_length, το batch size, το gradient accumulation, το learning rate και τις ρυθμίσεις αξιολόγησης. Με αυτές τις παραμέτρους φόρτωνε το μοντέλο και τον tokenizer, προετοίμαζε τα δεδομένα, εφάρμοζε LoRA ή QLoRA και στη συνέχεια εκτελούσε την εκπαίδευση μέσω του Hugging Face Trainer.

Στην περίπτωση της QLoRA, το βασικό μοντέλο φορτωνόταν με 4-bit quantization, NF4, double quantization και paged optimizer, με στόχο τη μείωση της VRAM. Στη LoRA FP16, αντίστοιχα, το μοντέλο φορτωνόταν σε fp16. Και στις δύο περιπτώσεις οι κύριες παράμετροι

του μοντέλου παρέμεναν παγωμένες και ενημερώνονταν μόνο οι LoRA παράμετροι, δηλαδή οι low-rank ενημερώσεις.

Το ίδιο script κατέγραφε τις βασικές μετρικές, όπως train loss, eval loss, perplexity, token-level accuracy, χρόνο εκπαίδευσης και peak VRAM. Επιπλέον, αποθήκευε το ιστορικό εκπαίδευσης, τον πίνακα αποτελεσμάτων, τις καμπύλες loss όπου υπήρχαν διαθέσιμα δεδομένα και ενδεικτικές απαντήσεις του μοντέλου σε σταθερά ερωτήματα ποιοτικού ελέγχου.

A.3 Post-processing (postprocess_all_runs.py)

Το script postprocess_all_runs.py χρησιμοποιήθηκε για τη συγκέντρωση και τον καθαρισμό των αποτελεσμάτων μετά την ολοκλήρωση των επιμέρους πειραμάτων. Ο ρόλος του ήταν να διατρέχει τους φακέλους των εκτελέσεων, να εντοπίζει τα αρχεία αποτελεσμάτων και να ενοποιεί τις βασικές μετρικές σε μορφή κατάλληλη για συγκριτική ανάλυση.

Η χρήση ξεχωριστού post-processing βήματος επέτρεψε την παραγωγή συγκεντρωτικών πινάκων και γραφημάτων χωρίς να χρειάζεται επανάληψη της εκπαίδευσης. Με αυτόν τον τρόπο, τα πειραματικά αποτελέσματα μπορούσαν να αναλυθούν με ενιαίο τρόπο ανά μοντέλο, μέθοδο, rank, dataset και υπολογιστικό περιβάλλον.

A.4 Συγκεντρωτικός πίνακας (make_master_results.py)

Το script make_master_results.py αξιοποιήθηκε για τη δημιουργία συγκεντρωτικού πίνακα αποτελεσμάτων από τα επιμέρους results_table.csv και metrics.json αρχεία. Ο συγκεντρωτικός πίνακας αποτέλεσε τη βάση για τους πίνακες και τα γραφήματα του Κεφαλαίου 4.

Η κοινή μορφή των αποτελεσμάτων ήταν σημαντική, επειδή οι εκτελέσεις έγιναν σε διαφορετικά περιβάλλοντα, όπως RTX 3070, Colab L4 και Colab A100. Με τα πεδία run_id και canonical_run_id, κάθε run μπορούσε να αντιστοιχιστεί με σαφήνεια στο αντίστοιχο μοντέλο, dataset, μέθοδο και rank. Έτσι, η ανάλυση παρέμεινε αναπαραγώγιμη και μειώθηκε ο κίνδυνος ασυνεπειών κατά τη μεταφορά των αποτελεσμάτων στους τελικούς πίνακες.

Συνολικά, τα scripts του συνοδευτικού υλικού ακολουθούν μια απλή και αναπαραγώγιμη πειραματική ροή: ο master runner ορίζει τις εκτελέσεις, το runner script πραγματοποιεί την εκπαίδευση και την αξιολόγηση, το post-processing συγκεντρώνει τα επιμέρους αποτελέσματα και το τελικό script δημιουργεί τον ενιαίο πίνακα που χρησιμοποιείται στην

ανάλυση. Η δομή αυτή διατηρεί τη διαδικασία διαφανή, χωρίς να επιβαρύνει το κυρίως σώμα της διατριβής με εκτενή πηγαίο κώδικα.

Παράρτημα Β: Ενδεικτικές εντολές εκτέλεσης και αναπαραγωγής

Στο παράρτημα παρατίθενται ενδεικτικές εντολές, όπως χρησιμοποιήθηκαν στο project, για αναπαραγωγή baseline, generalization και DAPT runs. Οι εντολές αντιστοιχούν στα scripts του Παραρτήματος Α.

B.1 Εκτέλεση run-set μέσω master runner

```
# Παράδειγμα (Colab): εκτέλεση προκαθορισμένου run-set
python THESIS_MASTER_RUNS.py --set l4_rep --script
"/content/drive/MyDrive/Colab
Notebooks/Thesis/scripts/run_peft_experiment.LOCAL.colab.py"
```

B.2 Baseline fine-tuning (dataset_v1)

```
python run_peft_experiment.LOCAL.colab.py --run_id auto --model
TinyLlama/TinyLlama-1.1B-Chat-v1.0 --method qlora --dataset dataset_v1
--r 16 --target_modules q_proj,v_proj --epochs 1 --max_length 256 -
-batch 1 --grad_accum 8 --lr 2e-4 --eval_test_size 0.05 --
eval_max_samples 500
```

B.3 Generalization evaluation (προαιρετικό)

```
# Train σε dataset_v1 και eval γενίκευσης σε dataset_wiki_50k
python run_peft_experiment.LOCAL.colab.py --run_id
gen_l4_tiny_v1_to_wiki_qlora_r16 --model TinyLlama/TinyLlama-1.1B-Chat-
v1.0 --method qlora --dataset dataset_v1 --generalization_dataset
dataset_wiki_50k --r 16 --target_modules q_proj,v_proj --epochs 1 -
-max_length 256 --batch 1 --grad_accum 8 --lr 2e-4 --eval_test_size
0.05 --eval_max_samples 500
```

B.4 Reverse generalization (train: dataset_wiki_50k → eval: dataset_v1)

```
python run_peft_experiment.LOCAL.colab.py --run_id
gen_l4_tiny_wiki_to_v1_qlora_r16_rev2 --model TinyLlama/TinyLlama-1.1B-
Chat-v1.0 --method qlora --dataset dataset_wiki_50k --
generalization_dataset dataset_v1 --r 16 --target_modules q_proj,v_proj
--epochs 1 --max_length 256 --batch 1 --grad_accum 8 --lr 2e-4 --
eval_test_size 0.05 --eval_max_samples 500
```

B.5 DAPT (continued pretraining-like) και fine-tuning

Παρατήρηση: Στις τελικές εκτελέσεις DAPT χρησιμοποιήθηκε προκαθορισμένος αριθμός βημάτων εκπαίδευσης. Το Stage-1 εκτελέστηκε στο dataset_pretrain_el για 3500 steps με max_train_samples=60000, ενώ το Stage-2 εκτελέστηκε στο dataset_v1 για 1500 steps, ξεκινώντας από τον adapter που είχε παραχθεί στο Stage-1.

```
# Stage-1: DAPT σε dataset_pretrain_el
```

```
python run_peft_experiment.LOCAL.colab.py --run_id  
dapt_l4_tiny_wikiel_qlora_r16_len128_s3500_b2 --model TinyLlama/TinyLlama-  
1.1B-Chat-v1.0 --method qlora --dataset dataset_pretrain_el --r 16 --  
target_modules q_proj,v_proj --max_length 128 --batch 2 --grad_accum 4 --lr  
2e-4 --eval_test_size 0.05 --eval_max_samples 200 --max_train_samples 60000  
--max_steps 3500 --generalization_dataset dataset_v1 --  
generalization_dataset dataset_wiki_50k --generalization_max_samples 200 --  
skip_generate
```

```
# Stage-2: fine-tuning σε dataset_v1 από τον adapter του Stage-1
```

```
python run_peft_experiment.LOCAL.colab.py --run_id  
dapt2_l4_tiny_wikiel_to_v1_qlora_r16_len128_b2 --model TinyLlama/TinyLlama-  
1.1B-Chat-v1.0 --method qlora --dataset dataset_v1 --r 16 --target_modules  
q_proj,v_proj --max_length 128 --batch 2 --grad_accum 4 --lr 2e-4 --  
eval_test_size 0.05 --eval_max_samples 50 --max_steps 1500 --  
init_adapter_path "/content/drive/MyDrive/Colab  
Notebooks/Thesis/outputs/dapt_l4_tiny_wikiel_qlora_r16_len128_s3500_b2/lora  
_adapter" --generalization_dataset dataset_wiki_50k --  
generalization_max_samples 500 --skip_generate
```

B.6 Ρυθμίσεις σταθερότητας (προαιρετικά)

```
# Σε περιπτώσεις fragmentation μνήμης (Colab):  
export PYTORCH_CUDA_ALLOC_CONF=expandable_segments:True
```

Παράρτημα Γ: Δομή φακέλων αποτελεσμάτων και παραγόμενα αρχεία

Το παράρτημα περιγράφει τη δομή των φακέλων αποτελεσμάτων (όπως παραδίδονται στο Data.zip) και τα βασικά artifacts ανά run.

Γ.1 Δομή φακέλων

```
Data/outputs/  
  rtx3070/  
    <run_id>/  
      results/  
        metrics.json  
        results_table.csv  
        log_history.csv  
        loss_curve.png  
        generations.json  
  colab_l4/  
    rep/<run_id>/results/...  
    generalization/<run_id>/results/...  
    dapt/<run_id>/results/...  
  colab_a100/  
    rep/<run_id>/results/...  
    capability/<run_id>/results/...  
  
Data/archive/  
  <run_id>/results/...
```

Γ.2 Περιγραφή βασικών αρχείων

metrics.json: σύνοψη μετρικών (eval_loss, perplexity, token_accuracy, train_runtime_sec, peak_vram_mib) και όπου ισχύει πληροφορία γενίκευσης (generalization).

results_table.csv: μία γραμμή ανά run, για εύκολη συγχώνευση/ενοποίηση πολλών runs.

log_history.csv: ιστορικό logging του Trainer (π.χ. loss ανά step).

loss_curve.png: γράφημα training loss που παράγεται από το log_history.csv (βλ. postprocess_all_runs.py).

generations.json: ποιοτικά δείγματα εξόδου (όπου ενεργοποιείται η παραγωγή).

Υπεύθυνη Δήλωση Συγγραφέα

Δηλώνω ρητά ότι, σύμφωνα με το άρθρο 8 του Ν.1599/1986, η παρούσα εργασία αποτελεί αποκλειστικά προϊόν προσωπικής μου εργασίας, δεν προσβάλλει κάθε μορφής δικαιώματα διανοητικής ιδιοκτησίας, προσωπικότητας και προσωπικών δεδομένων τρίτων, δεν περιέχει έργα/εισφορές τρίτων για τα οποία απαιτείται άδεια των δημιουργών/δικαιούχων και δεν είναι προϊόν μερικής ή ολικής αντιγραφής. Οι πηγές που χρησιμοποιήθηκαν παρατίθενται αποκλειστικά στο κεφάλαιο της βιβλιογραφίας.