

ΠΑΝΕΠΙΣΤΗΜΙΟ ΜΑΚΕΔΟΝΙΑΣ
ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ
ΤΜΗΜΑΤΟΣ ΕΦΑΡΜΟΣΜΕΝΗΣ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΜΗΧΑΝΙΚΗ ΕΝΤΟΛΩΝ ΓΙΑ ΤΗΝ ΑΠΟΔΟΤΙΚΗ ΧΡΗΣΗ ΜΟΝΤΕΛΩΝ GPT ΜΕ
ΣΤΟΧΟ ΤΗΝ ΠΑΡΑΓΩΓΗ ΛΟΓΙΣΜΙΚΟΥ

Διπλωματική Εργασία

του

Θεόδωρου Ιωαννίδη

Θεσσαλονίκη, Φεβρουάριος 2024

ΜΗΧΑΝΙΚΗ ΕΝΤΟΛΩΝ ΓΙΑ ΤΗΝ ΑΠΟΔΟΤΙΚΗ ΧΡΗΣΗ ΜΟΝΤΕΛΩΝ GPT ΜΕ
ΣΤΟΧΟ ΤΗΝ ΠΑΡΑΓΩΓΗ ΛΟΓΙΣΜΙΚΟΥ

Θεόδωρος Ιωαννίδης

Πτυχίο Εφαρμοσμένης Πληροφορικής ΠΑ.ΜΑΚ., 2023

Διπλωματική Εργασία

υποβαλλόμενη για τη μερική εκπλήρωση των απαιτήσεων του

ΜΕΤΑΠΤΥΧΙΑΚΟΥ ΤΙΤΛΟΥ ΣΠΟΥΔΩΝ ΣΤΗΝ ΕΦΑΡΜΟΣΜΕΝΗ
ΠΛΗΡΟΦΟΡΙΚΗ

Επιβλέπων Καθηγητής
Αλέξανδρος Χατζηγεωργίου

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την ηη/μμ/εεεε

Χατζηγεωργίου Αλέξανδρος

Σατρατζέμη Μαρία

Ξυνόγαλος Στυλιανός

.....

.....

.....

Θεόδωρος Ιωαννίδης

.....

Περίληψη

Η τεχνητή νοημοσύνη και πιο συγκεκριμένα τα μεγάλα γλωσσικά μοντέλα έχουν δυνατότητες να προσφέρουν υποστήριξη σε δημιουργικές εργασίες όπως η παραγωγή κώδικα. Στην εργασία αυτή γίνεται μία απόπειρα εντοπισμού, ανάλυσης και αξιοποίησης των μεθόδων με τις οποίες ο κάθε ενδιαφερόμενος δύναται να χρησιμοποιήσει αυτά τα εργαλεία με έμφαση στην παραγωγή κώδικα. Η εργασία αρχικά πραγματεύεται την υπάρχουσα βιβλιογραφία στο πεδίο αυτό, όπου ερευνώνται τεχνικές όπως οι Chain of Thought και Few Shot καθώς και πληθώρα καλών πρακτικών και συστάσεων, ενώ δίνεται ιδιαίτερη βαρύτητα στην συγγραφή κώδικα μέσω αυτής της προσέγγισης. Έπειτα, παρουσιάζονται και σχολιάζονται πολλά από τα διαθέσιμα εργαλεία υποβοήθησης για prompt engineering και prompt generation. Αξιοποιώντας τα πορίσματα αυτής της έρευνας υλοποιείται η πρακτική συνεισφορά της εργασίας στο αντικείμενο. Έτσι δημιουργήθηκαν και παρουσιάζονται αναλυτικά, μία ιστοσελίδα η οποία παρέχει υπηρεσίες prompt engineering και ένα extension στο προγραμματιστικό περιβάλλον Visual Studio Code το οποίο ενσωματώνει άμεση επικοινωνία με το μοντέλο GPT 3.5 της OpenAI. Στόχοι αυτών των εφαρμογών είναι η περαιτέρω αξιοποίηση των μεγάλων γλωσσικών μοντέλων πέρα από τον συνήθη τρόπο χρήσης τους, η αποδοτικότερη εκμετάλλευσή και ενσωμάτωσή τους σε ολοκληρωμένα συστήματα και η προσαύξηση της παραγωγικότητάς τους.

Λέξεις Κλειδιά: Μηχανική εντολών, Μεγάλα γλωσσικά μοντέλα, Τεχνητή νοημοσύνη, κώδικας

Abstract

Artificial intelligence and more specifically Large Language Models make useful companions for creative works such as writing code. This thesis is an attempt to identify, analyze and make use of the methods which any willing person could leverage with respect to writing code. This thesis first examines current bibliography in the field where techniques such as Chain of Thought and Few Shot are being studied, alongside a variety of good practices and recommendations with extra focus on applying this approach for writing code. Then many of the available tools for prompt engineering and prompt generation are being presented and examined. Utilizing the findings of this study, the contribution of this thesis to the field is materialized. A website offering prompt engineering services and an extension for the development environment Visual Studio Code which features direct access to the GPT 3.5 model by OpenAI were developed and thoroughly presented. The goals of these applications are to extend the utilization of Large Language Models beyond their usual way of being used, the more effective exploitation and integration of such tools into complete systems and the increase of their productivity.

Keywords: prompt, prompt engineering, Large Language Models, Artificial intelligence, coding

Πρόλογος – Ευχαριστίες

Τα μεγάλα γλωσσικά μοντέλα είναι συναρπαστικά εργαλεία με εκπληκτικές δυνατότητες και προοπτικές τις οποίες θαύμασα από την πρώτη στιγμή που τα συνάντησα και θέλησα να ασχοληθώ μαζί τους. Αυτή η εργασία συνεχίζει σε αυτή την κατεύθυνση.

Θα ήθελα να ευχαριστήσω όλους όσους βοήθησαν στην ολοκλήρωση αυτού του έργου, τον κύριο Χατζηγεωργίου και την οικογένειά μου για την συνεχή τους υποστήριξη και για όλα όσα προσέφεραν.

Περιεχόμενα

| | | |
|--------|--|----|
| 1 | Εισαγωγή | 1 |
| 1.1 | Πρόβλημα – Σημαντικότητα του θέματος | 1 |
| 1.2 | Σκοπός – Στόχοι | 2 |
| 1.3 | Ερωτήματα – Υποθέσεις | 3 |
| 1.4 | Συνεισφορά | 4 |
| 1.5 | Βασική Ορολογία | 4 |
| 1.6 | Διάρθρωση της μελέτης | 5 |
| 2 | Βιβλιογραφική Επισκόπηση – Θεωρητικό Υπόβαθρο | 6 |
| 2.1 | Γνωστές τεχνικές και πρακτικές | 6 |
| 2.1.1 | Prompt based learning | 6 |
| 2.1.2 | Chain of Thought | 8 |
| 2.1.3 | Few shot | 12 |
| 2.1.4 | Άλλες τεχνικές prompting | 15 |
| 2.1.5 | Metaprompts | 17 |
| 2.1.6 | Καλές πρακτικές και συστάσεις | 17 |
| 2.1.7 | Συχνά λάθη στη συγγραφή prompts | 21 |
| 2.1.8 | Πρότυπα prompting | 22 |
| 2.1.9 | Τεχνικές prompting για τη συγγραφή κώδικα | 24 |
| 2.1.10 | Αποσφαλμάτωση μέσω LLMs | 33 |
| 2.1.11 | Η σημασία ενός περιβάλλοντος φιλικού προς τον χρήστη | 35 |
| 2.2 | Εργαλεία αρωγής στη μηχανική εντολών | 36 |
| 2.2.1 | Γενική προσέγγιση | 36 |
| 2.2.2 | Εργαλεία prompt generation | 39 |
| 2.2.3 | Εργαλεία prompt engineering | 40 |
| 3 | Μεθοδολογία | 50 |
| 3.1 | Παρουσίαση των εργαλείων | 51 |
| 3.2 | Η υλοποίηση του ιστοτόπου | 52 |
| 3.2.1 | Η υλοποίηση της σελίδας “Advanced” | 52 |
| 3.2.2 | Η υλοποίηση της σελίδας “Begginers” | 57 |
| 3.3 | Η υλοποίηση του extension | 63 |
| 3.4 | Η πρακτική χρησιμότητα της ιστοσελίδας | 69 |

| | |
|--|----|
| 3.5 Η πρακτική χρησιμότητα του extension | 73 |
| 4 Περιήγηση στις υλοποιήσεις | 75 |
| 4.1 Περιήγηση στη σελίδα Advanced | 75 |
| 4.2 Περιήγηση στη σελίδα Begginers | 78 |
| 4.3 Περιήγηση στο extension | 80 |
| 5 Αξιολόγηση των υλοποιήσεων | 85 |
| 5.1 Αξιολόγηση της ιστοσελίδας | 85 |
| 6 Επίλογος | 92 |
| 6.1 Σύνοψη και συμπεράσματα | 92 |
| 6.2 Όρια και περιορισμοί της έρευνας | 95 |
| 6.3 Μελλοντικές Επεκτάσεις | 96 |

Κατάλογος Εικόνων

| | |
|--|----|
| Εικόνα 1: Το interface της σελίδας "Advanced" | 54 |
| Εικόνα 2: Το interface της σελίδας "Beginners" | 59 |
| Εικόνα 3: Παράδειγμα χρήσης του extension | 64 |
| Εικόνα 4: Guide through στο extension | 80 |
| Εικόνα 5: Guide through στο extension | 81 |
| Εικόνα 6: Guide through στο extension | 82 |
| Εικόνα 7: Guide through στο extension | 83 |
| Εικόνα 8: Διαγράμματα αξιολόγησης | 85 |
| Εικόνα 9: Διαγράμματα αξιολόγησης | 86 |
| Εικόνα 10: Διαγράμματα αξιολόγησης | 87 |
| Εικόνα 11: Διαγράμματα αξιολόγησης | 88 |
| Εικόνα 12: Διαγράμματα αξιολόγησης | 89 |

Συμβολισμοί

LLM: Large language Model

AI: Artificial Intelligence

CoT: Chain of Thought

GPT: Generative Pre-Trained Transformer

1 Εισαγωγή

1.1 Πρόβλημα – Σημαντικότητα του θέματος

Τα Large Language Models (LLMs) και τα Generative Pre-trained Transformers (GPT), είναι ένας τύπος τεχνητής νοημοσύνης τα οποία, πιο συγκεκριμένα υλοποιούνται από τεχνητά νευρωνικά δίκτυα (ANN) που έχουν εκπαιδευτεί σε ένα πολύ μεγάλο σύνολο γλωσσικών δεδομένων κειμένου και κώδικα. Αυτό τους επιτρέπει να παράγουν κείμενο, να απαντούν σε προβλήματα, να μεταφράζουν γλώσσες, να γράφουν διαφορετικά είδη δημιουργικού περιεχομένου και να απαντούν σε πάσης φύσεως ερωτήσεις με τρόπο παρόμοιο με τον ανθρώπινο αλλά και να παράγουν κώδικα για λογισμικό, πάντα σύμφωνα με τις οδηγίες του χρήστη.

Τα LLMs είναι ακόμα υπό ανάπτυξη και εν πολλοίς σε πειραματικό στάδιο, αλλά έχουν ήδη τη δυνατότητα να αλλάξουν δραματικά τον τρόπο με τον οποίο ζούμε και εργαζόμαστε. Για παράδειγμα, μπορούν να χρησιμοποιηθούν για την αυτοματοποίηση εργασιών που σήμερα γίνονται από ανθρώπους, όπως η συγγραφή άρθρων, η μετάφραση εγγράφων και φυσικά η ανάπτυξη λογισμικού. Μπορούν επίσης να χρησιμοποιηθούν για την ανάπτυξη νέων προϊόντων και υπηρεσιών, όπως chatbots που μπορούν να παρέχουν υποστήριξη στους πελάτες ή εκπαιδευτικά εργαλεία που μπορούν να βοηθήσουν τους μαθητές στην εκπαιδευτική διαδικασία.

Το prompting είναι μία τεχνική που χρησιμοποιείται για να βοηθήσει τα μοντέλα GPT να κατανοήσουν τι τους ζητείται να κάνουν και να παράγουν το επιθυμητό αποτέλεσμα. Το prompt είναι ένα μικρό κομμάτι κειμένου το οποίο δίνεται στο μοντέλο προκειμένου να του παρέχει πληροφορίες σχετικά με το είδος, το ύφος και το θέμα του περιεχομένου το οποίο θα πρέπει να παράγει. Τα prompts είναι συνήθως σύντομα και περιεκτικά και η σωστή διατύπωσή τους μπορεί να βοηθήσει το LLM να παράγει πιο ακριβές και εύστοχο περιεχόμενο [1].

Για παράδειγμα, εάν ζητήσουμε από ένα LLM να γράψει ένα ποίημα για την αγάπη, μπορούμε να του δώσουμε ένα prompt όπως "Γράψτε ένα ποίημα για την αγάπη".

1 Εισαγωγή

Το prompt θα βοηθήσει το LLM να κατανοήσει ότι πρόκειται να γράψει ένα ποίημα για την αγάπη και θα το βοηθήσει να επιλέξει τις σωστές λέξεις και φράσεις για να δημιουργήσει ένα ποίημα που είναι σχετικό με το θέμα.

Το prompt engineering είναι ένα σύνολο τεχνικών που χρησιμοποιούνται για τη βελτίωση της απόδοσης των LLMs. Ο σκοπός του prompt engineering είναι να βοηθήσει τα LLMs να παράγουν πιο ακριβές, σχετικό και δημιουργικό περιεχόμενο. Ο στόχος αυτός επιτυγχάνεται με την ακριβέστερη διατύπωση του προβλήματος, με την παροχή παραδειγμάτων του επιθυμητού αποτελέσματος, με την πιο διεξοδική περιγραφή της επιθυμητής εργασίας κ.ά. Με τη χρήση του prompt engineering, μπορούμε να βοηθήσουμε τα LLMs να κατανοήσουν καλύτερα την εργασία που τα καλούμε να εκτελέσουν και να παράγουν αποτελέσματα που είναι πιο κοντά στο ζητούμενο. Το prompt engineering είναι ένα νέο και δυναμικά εξελισσόμενο πεδίο έρευνας. Καθώς η έρευνα συνεχίζεται, αναμένουμε να δούμε ακόμη πιο βελτιωμένη και αποδοτική χρήση των LLMs.

1.2 Σκοπός – Στόχοι

Σκοπός της εργασίας αυτής είναι η καταγραφή, η ανάλυση, η εξέταση και ο εμπειρισματομένος σχολιασμός των τεχνικών που χρησιμοποιούνται για την εφαρμογή του prompt engineering. Παρότι το prompt engineering είναι μία καινούργια και συνεχώς εξελισσόμενη τεχνική υπάρχουν ήδη κάποιες μεθοδολογίες οι οποίες έχουν ευρύτατη χρήση και μας βοηθάνε στο να γράψουμε καλύτερα prompts τα οποία είναι πιο κατανοητά και επιφέρουν τα επιθυμητά αποτελέσματα. Υπάρχει ήδη αρκετή βιβλιογραφία πάνω στον τομέα του prompt engineering με διαφορετικά άρθρα να εστιάζουν είτε στην συγγραφή κώδικα είτε στη συγγραφή κειμένου είτε στην εκτέλεση άλλων εργασιών μέσω των Large Language Models με σκοπό την παραγωγή κώδικα λογισμικού. Παράλληλα στόχος της εργασίας είναι και η εμπειρική μελέτη του θέματος και η εξαγωγή κατευθυντήριων γραμμών αλλά και μεθόδων αξιολόγησης των παραγόμενων λύσεων.

Επόμενο βήμα είναι η σύνθεση και ανάπτυξη μιας ολοκληρωμένης και αποτελεσματικής προσέγγισης στο ζήτημα της συγγραφής κατάλληλων prompts για την

παραγωγή κώδικα. Φαίνεται να υπάρχουν σημαντικές διαφοροποιήσεις στις προσεγγίσεις των prompts ανάλογα με την εκάστοτε εργασία που θέλουμε να εκτελέσουμε. Οι τεχνικές αυτές μπορεί να αποσκοπούν είτε στο να δώσουν στο εργαλείο να κατανοήσει καλύτερα την εργασία την οποία καλείται να εκτελέσει είτε να κατανοήσει καλύτερα τα βήματα τα οποία θα πρέπει να ακολουθήσει προκειμένου να οδηγηθεί στο επιθυμητό αποτέλεσμα. Ακόμα μία δημοφιλής και πολλά υποσχόμενη τεχνική είναι η παροχή στο εργαλείο παραδειγμάτων των σωστών αποτελεσμάτων ούτως ώστε εξετάζοντας τα να αποκτήσει μία καλή εικόνα του ποιο είναι το σωστό αποτέλεσμα στο οποίο θα πρέπει να καταλήξει.

Η ελλιπής επεξήγηση της επιθυμητής εργασίας, η ελλιπής επεξήγηση των τμημάτων στα οποία αυτή συνιστάται ή η ελλιπής παράθεση παραδειγμάτων του σωστού αποτελέσματος αποτελούν κάποια από τα σημαντικότερα λάθη τα οποία οδηγούν σε κακή απόδοση αυτών των εργαλείων. Ένα εργαλείο τεχνητής νοημοσύνης για να κατανοήσει την εργασία την οποία του ζητάμε να εκτελέσει και να οδηγηθεί στο σωστό αποτέλεσμα είναι απαραίτητο να διαθέτει μία πολύ καλή αντίληψη των βημάτων αυτής της εργασίας, του πως να τα εκτελέσει αλλά και σε ποια μορφή θα πρέπει να είναι το επιθυμητό αποτέλεσμα. Εάν αυτές οι πληροφορίες λείπουν, ειδικότερα σε πιο περίπλοκες εργασίες, το εργαλείο τεχνητής νοημοσύνης έχοντας ελλιπή αντίληψη της εργασίας δεν θα μπορέσει να ακολουθήσει τα σωστά βήματα τα οποία θα ακολουθούσε ένας άνθρωπος ο οποίος αντιλαμβάνεται πλήρως τις ανάγκες και τις απαιτήσεις της εργασίας και ως εκ τούτου θα οδηγηθεί σε ένα αποτέλεσμα το οποίο δεν ανταποκρίνεται στις ανθρώπινες προσδοκίες. Επιπλέον, θα επιχειρηθεί η ανάπτυξη γραφικής διεπαφής χρήστη σε μορφή ιστοσελίδας, για τον καθορισμό βασικών παραμέτρων του υπό ανάπτυξη συστήματος, η αλληλεπίδραση με πλατφόρμα τεχνητής νοημοσύνης (π.χ. chatGPT, Bard, Copilot κτλ) και η παρακολούθηση των αποτελεσμάτων. Επίσης θα επιχειρηθεί η ενσωμάτωση ενός αντίστοιχου είδους εργαλείου σε προγραμματιστικό περιβάλλον.

1.3 Ερωτήματα – Υποθέσεις

1 Εισαγωγή

Τα ερωτήματα αυτής της έρευνας σχετίζονται με το πως, σε ποιον βαθμό και τίνι τρόπο δύνανται τα Μεγάλα Γλωσσικά Μοντέλα τύπου GPT να επιτελέσουν δημιουργικές εργασίες με έμφαση στην παραγωγή κώδικα προγραμματισμού. Προτού επιχειρήσουμε να απαντήσουμε αυτά τα ερωτήματα θεωρούμε πως αυτά τα εργαλεία έχουν προοπτικές να προσφέρουν σε αυτόν τον τομέα και πως υπάρχει νόημα στο να ερευνήσουμε τις μεθόδους βέλτιστης αξιοποίησής τους από τους ανθρώπους.

1.4 Συνεισφορά

Στα πλαίσια αυτής της μελέτης αρχικά ερευνήθηκε η υπάρχουσα βιβλιογραφία η οποία πραγματεύεται τη χρήση μεγάλων γλωσσικών μοντέλων σε δημιουργικές εργασίες όπως η συγγραφή κώδικα. Έπειτα ερευνώντας τις κατευθύνσεις και τις συστάσεις οι οποίες δίνονται στη βιβλιογραφία, εντοπίσαμε και συγκεντρώσαμε κάποιες καλές πρακτικές που βοηθούν στην αποδοτικότερη χρήση αυτών των εργαλείων καθώς και κάποια συχνά λάθη τα οποία θα πρέπει να αποφεύγονται.

Σε επόμενο στάδιο αξιοποιήθηκαν τα παραπάνω πορίσματα προκειμένου να υλοποιηθεί μία ιστοσελίδα με λειτουργίες prompt engineering με σκοπό την αποδοτικότερη αξιοποίηση των μοντέλων GPT-3,5 και GPT-4 στην παραγωγή και αποσφαλμάτωση κώδικα. Επιπλέον δημιουργήθηκε ένα extension για το προγραμματιστικό περιβάλλον Visual Studio Code το οποίο αξιοποιεί το μοντέλο GPT-3,5 για την υποστήριξη των προγραμματιστών καθ' όλη την διαδικασία παραγωγής λογισμικού.

1.5 Βασική Ορολογία

Τεχνητή νοημοσύνη: ένα πρόγραμμα λογισμικού το οποίο επιτελεί συγκεκριμένες εργασίες χωρίς να έχει λάβει σαφείς οδηγίες, αξιοποιώντας αλγορίθμους οι οποίοι του παρέχουν τις κατάλληλες κατευθύνσεις για την εκτέλεση εργασιών αυτού του τύπου.

Μεγάλο γλωσσικό μοντέλο: ένα είδος τεχνητής νοημοσύνης, πιο συγκεκριμένα, ένα νευρωνικό δίκτυο το οποίο δέχεται ως είσοδο κείμενο σε φυσική γλώσσα και επιστρέφει μία απάντηση επίσης σε φυσική γλώσσα.

GPT: Generative Pre-trained Transformer. Με απλά λόγια, αποτελεί έναν τύπο υλοποίησης Μεγάλου Γλωσσικού Μοντέλου χρήσει αρχιτεκτονικής «transformer» εκπαιδευμένο σε μεγάλα σύνολα unlabeled δεδομένων.

1.6 Διάρθρωση της μελέτης

Στο κεφάλαιο 2 παρουσιάζονται τα ευρήματα της βιβλιογραφικής έρευνας σχετικά με την αποτελεσματικότητα της χρήσης μοντέλων τύπου GPT σε διάφορες εργασίες με έμφαση στην παραγωγή κώδικα καθώς και τα αντίστοιχα πορίσματα σχετικά με τις καλές πρακτικές, τις συστάσεις και τη μεθοδολογία της διαδικασίας. Στο κεφάλαιο 3 παρουσιάζονται οι υλοποιήσεις της εργασίας, ήτοι, η ιστοσελίδα με λειτουργίες prompt engineering και το extension για το Visual Studio Code, ο τρόπος λειτουργίας τους καθώς και η πρακτική τους χρησιμότητα. Στο κεφάλαιο 4 θα δούμε μία περιήγηση σε πραγματικά σενάρια χρήσης των υλοποιήσεων. Στο κεφάλαιο 5 παρουσιάζονται τα αποτελέσματα της αξιολόγησης των υλοποιήσεων από πραγματικούς χρήστες. Τέλος, στο κεφάλαιο 6 αναπτύσσονται διάφοροι προβληματισμοί και σκέψεις σχετικά με την έρευνα καθώς και κάποιες σκέψεις και φιλοδοξίες για το μέλλον.

2 Βιβλιογραφική Επισκόπηση – Θεωρητικό Υπόβαθρο

Παρότι το prompt engineering είναι μία καινούργια και συνεχώς εξελισσόμενη τεχνική υπάρχουν ήδη κάποιες μεθοδολογίες οι οποίες έχουν ευρύτατη χρήση και μας βοηθάνε στο να γράψουμε καλύτερα prompts τα οποία είναι πιο κατανοητά και επιφέρουν τα επιθυμητά αποτελέσματα. Υπάρχει ήδη αρκετή βιβλιογραφία πάνω στον τομέα του prompt engineering με διαφορετικά άρθρα να εστιάζουν είτε στην συγγραφή κώδικα είτε στη συγγραφή κειμένου είτε στην εκτέλεση άλλων εργασιών. Φαίνεται να υπάρχουν σημαντικές διαφοροποιήσεις στις προσεγγίσεις των prompts ανάλογα με την εκάστοτε εργασία που θέλουμε να εκτελέσουμε. Οι τεχνικές αυτές μπορεί να αποσκοπούν είτε στο να δώσουν στο εργαλείο να κατανοήσει καλύτερα την εργασία την οποία καλείται να εκτελέσει είτε να κατανοήσει καλύτερα τα βήματα τα οποία θα πρέπει να ακολουθήσει προκειμένου να οδηγηθεί στο επιθυμητό αποτέλεσμα. Ακόμα μία δημοφιλής και πολλά υποσχόμενη τεχνική είναι η παροχή στο εργαλείο παραδειγμάτων σωστών αποτελεσμάτων ούτως ώστε εξετάζοντας τα να αποκτήσει μία καλή εικόνα του ποιο είναι το σωστό αποτέλεσμα στο οποίο θα πρέπει να καταλήξει.

Όσο οι χρήστες εξοικειώνονται με την έννοια του prompt engineering είναι πολύ πιθανό να αρχίσουν να αντικαθιστούν την αναζήτηση στο Internet με το prompting σε Large Language Models. Για παράδειγμα αντί να αναζητήσουν ποια είναι την κατάλληλη βιβλιοθήκη για μία εργασία θα μπορούσαν να απλώς να ζητήσουν από το μοντέλο να εισάγει την κατάλληλη βιβλιοθήκη [2].

2.1 Γνωστές τεχνικές και πρακτικές

2.1.1 *Prompt based learning*

Όπως ένας άνθρωπος μπορεί να εκπαιδευτεί σε μία διαδικασία ακούγοντας τις οδηγίες για την διεκπεραίωση της καθώς και παρατηρώντας κάποια παραδείγματα έτσι και σε ένα νευρωνικό δίκτυο οι οδηγίες μιας εργασίας παρέχουν μία επιπλέον διάσταση εκπαίδευσης συχνά σε πιο αφαιρετικό αλλά και εύληπτο επίπεδο. Ειδικότερα οι διεργασίες όπου τα παραδείγματα για pre training είναι λιγοστά, η πιο πρακτική προσέγγιση είναι να χτίσουμε συστήματα βάσει οδηγιών και χειρόγραφων

παραδειγμάτων. Μέσα από έρευνες έχει φανεί πως η στοχευμένη καθοδήγηση είναι το κλειδί για την επίτευξη cross task γενίκευσης αντί για την εκπαίδευση σε πολλά διαφορετικά είδη εργασιών [3].

Το σκεπτικό είναι το εξής να ενσωματώσουμε την πληροφορία του θέματος μέσα στην είσοδο με τη μορφή φυσικής γλώσσας η οποία έπειτα θα αξιοποιηθεί για την διενέργεια της εργασίας [4]. Η μέθοδος του prompt learning προτιμάται συγκριτικά με το fine-tuning διότι είναι πολύ λιγότερο κοστοβόρα καθώς επηρεάζει λιγότερες παραμέτρους [5]. Επίσης εν αντιθέσει με το fine-tuning δεν απαιτείται επιπλέον στρώμα στο νευρωνικό δίκτυο ενώ εξαιρετικές αποδόσεις επιτυγχάνονται ακόμα και με ελάχιστα παραδείγματα [6].

Οι προσεγγίσεις τύπου fine tuning απαιτούν τη χρήση ενός loss function προκειμένου να μεταβληθούν οι παράμετροι του μοντέλου, ωστόσο στο prompt based learning δεν υπάρχει training οπότε το μοντέλο παραμένει αμετάβλητο. Αυτό βέβαια συνοδεύεται και με το μειονέκτημα πως ο αριθμός των παραδειγμάτων που μπορούν να παρασχεθούν στο μοντέλο είναι ιδιαίτερα περιορισμένος. Αυτό αποτελεί ιδιαίτερο πρόβλημα σε εργασίες οι οποίες απαιτούν δομημένα δεδομένα. Μία ιδέα για την επίλυση αυτού του προβλήματος είναι ενσωμάτωση πιο αναλυτικών περιγραφών παρά η παροχή επιπλέον παραδειγμάτων [7].

Με μία διαδικασία η οποία μιμείται το pre training, το prompting γεμίζει το κενό ανάμεσα στο tuning και στο training. Αυτή η διαδικασία έχει αποδειχθεί ιδιαίτερα αποτελεσματική ακόμα και με ελάχιστα δεδομένα καθώς ακόμα και το zero shot prompt learning μπορεί να ξεπεράσει ακόμα και το 32 shot fine tuning. Ενώ όσον αφορά μεγαλύτερα μοντέλα η βελτιστοποίηση της εισόδου μπορεί να έχει παρόμοια επίδραση με ένα πλήρες fine-tuning των παραμέτρων[8]. Ακολουθώντας αυτά τα πορίσματα μπορούμε να σχηματίσουμε ζεύγη εισόδων και επιθυμητών εξόδων προκειμένου να τα παρέχουμε μαζί με τις οδηγίες στο μοντέλο ώστε να το εκπαιδύσουμε με δεδομένα σε μορφή παρόμοια με αυτήν της αρχικής του εκπαίδευσης αλλά με πιο άμεσο τρόπο.

2 Βιβλιογραφική Επισκόπηση – Θεωρητικό Υπόβαθρο

Τα μεγάλα pre trained γλωσσικά μοντέλα έχουν επιδείξει εκπληκτικές ικανότητες γενίκευσης σε άγνωστες περιπτώσεις, εκπαιδευόμενα μόνο και μόνο μέσα από λίγα παραδείγματα. Ωστόσο έχει φανεί πως η απόδοση εξαρτάται σε μεγάλο βαθμό από τα συγκεκριμένα παραδείγματα με αποτέλεσμα άλλες φορές να είναι άψογη και άλλες φορές να φαίνεται σχεδόν τυχαία. Για να μετριαστεί αυτό το φαινόμενο μπορούμε να επιλέγουμε παραδείγματα τα οποία να είναι όσο το δυνατόν πλησιέστερα σε επίπεδο semantics στην πραγματική περίπτωση ή να χρησιμοποιούμε για εκπαίδευση όσο το δυνατόν πιο περίπλοκα παραδείγματα [9].

Με την προσέγγιση τύπου prompt learning, αποσκοπούμε στο να εισάγουμε στο μοντέλο, νέες γνώσεις και να του παρέχουμε επιπλέον στοιχεία σχετικά με το αντικείμενο και το αποτέλεσμα στο οποίο θα πρέπει να καταλήξει. Για να επιτευχθεί αυτός ο στόχος, οι επιπλέον πληροφορίες, θα παρασχεθούν με την μορφή φυσικής γλώσσας, ενσωματωμένες στην είσοδο του εργαλείου. Είναι μία μέθοδος πιο πρακτική από το fine tuning καθώς μπορεί να υλοποιηθεί εύκολα, γρήγορα και φθηνά με την παράθεση μόνο λίγων παραδειγμάτων, χωρίς την παραμετροποίηση του μοντέλου. Παράλληλα, έχει φανεί πως προσφέρει εφάμιλλα αποτελέσματα με το fine tuning με μοναδικό προβληματισμό, την επιλογή εύστοχων και συμπεριληπτικών παραδειγμάτων καθότι, το περιορισμένο πλήθος τους καθιστά το μοντέλο ευαίσθητο στο καθένα από αυτά.

2.1.2 Chain of Thought

Μία πολύ σημαντική τεχνική η οποία μπορεί να βοηθήσει είναι το Chain of thought ή αλλιώς task decomposing δηλαδή η τμηματοποίηση της εργασίας σε μικρότερα sub tasks τα οποία έχουν υψηλότερη πιθανότητα να εκτελεστούν με επιτυχία. Το τελικό prompt αποτελείται από μία αλυσίδα μικρότερων sub tasks τα οποία παρέχουν μία αναλυτικότερη, πιο διεξοδική και πιο δομημένη περιγραφή της εργασίας με αποτέλεσμα αυτή να εκτελείται σε μικρά και εύκολα βήματα το καθένα από τα οποία διαθέτει υψηλότερες πιθανότητες επιτυχίας και εν τέλει είναι εύκολο να ολοκληρωθούν και να καταλήξουν στο αποτέλεσμα ολόκληρης της εργασίας. Το καθένα από αυτά τα sub tasks θα διερευνηθεί ξεχωριστά κάτι το οποίο προσφέρει διευρυμένες δυνατότητες επεξεργασίας, για το κάθε υποερώτημα ξεχωριστά και παρέχει στο μοντέλο τη

δυνατότητα να αξιοποιήσει πιο εύστοχα τις γνώσεις τις οποίες διαθέτει από την εκπαίδευσή του.

Ακόμα η τεχνική σε Chain of thought οδηγεί σε απαντήσεις οι οποίες είναι πιο κατανοητές και για τον χρήστη και για το μοντέλο κάτι το οποίο παρέχει στο μοντέλο την δυνατότητα να μας εξηγήσει πως κατέληξε σε μία συγκεκριμένη απάντηση ή ποια ήταν η λογική την οποία ακολούθησε και με αυτό τον τρόπο είναι και πιο εύκολη η διόρθωση τυχόν λανθασμένων απαντήσεων ή η τροποποίηση των βημάτων. Τα οφέλη της χρήσης αυτής της τεχνικής είναι περισσότερο εμφανή σε πιο περίπλοκα προβλήματα τα οποία είναι δύσκολο να γίνουν αντιληπτά και να επιλυθούν απευθείας και συνεπώς η προσέγγιση βήμα-βήμα είναι αναγκαία. Η βελτίωση στην απόδοση ενός μοντέλου με τη χρήση αυτής της τεχνικής είναι τόσο σημαντική που μπορεί ακόμα και να ξεπεράσει την απόδοση κάποιων fine-tuned task specific μοντέλων. Το γεγονός πως η ανάλυση της εργασίας σε υποεργασίες γίνεται με τη χρήση φυσικής γλώσσας σημαίνει πως η εφαρμογή αυτής της τεχνικής είναι δυνατή σε πολλά είδη προβλημάτων όπως μαθηματικά προβλήματα, προβλήματα λογικής, προγραμματισμού ή άλλα πιο εξειδικευμένα προβλήματα από τον οποιονδήποτε χρήστη. Έχει παρατηρηθεί πως η τεχνική Chain of thought έχει καλύτερα αποτελέσματα όσο πιο μεγάλο είναι το μέγεθος του μοντέλου [3].

Για παράδειγμα, εάν επιθυμούμε να λάβουμε μία παρουσίαση των διαφορών ανάμεσα στα ηλεκτρικά και στα βενζινοκίνητα αυτοκίνητα, μπορούμε να οργανώσουμε το αίτημα ως εξής: *«Θέλω να γράψεις μία παρουσίαση των διαφορών ανάμεσα στα ηλεκτρικά και στα βενζινοκίνητα αυτοκίνητα, ακολουθώντας τα παρακάτω βήματα.*

- 1. Γράψε κάποιες γενικές διαφορές*
- 2. Γράψε για τις διαφορές ανάμεσα στους κινητήρες*
- 3. Γράψε για τις διαφορές στα σχετικά κόστη*
- 4. Ανέφερε κάποια χαρακτηριστικά παραδείγματα από σύγχρονα οχήματα».*

Έτσι, αφενός θα πάρουμε μία απάντηση η οποία θα είναι πιο κοντά σε αυτό που έχουμε στο μυαλό μας και αφετέρου μειώνονται οι πιθανότητες να συμπεριληφθούν άσχετες πληροφορίες.

2 Βιβλιογραφική Επισκόπηση – Θεωρητικό Υπόβαθρο

Φαίνεται πως οι τεχνική Chain of thought είναι πολύ χρήσιμη και σε προβλήματα προγραμματισμού παρέχοντας βελτιωμένη απόδοση ειδικότερα σε προβλήματα τα οποία παραμένουν δυσεπίλυτα ακόμα και με τη χρήση μεγάλων μοντέλων [10].

Σε μελέτες που έχουν γίνει φαίνεται πως ο κώδικας ο οποίος παράγεται από το γλωσσικό μοντέλο κώδικα Copilot του Github έχει υψηλά ποσοστά εγκυρότητας τα οποία ξεπερνούν το 90% αποδεικνύοντας την ικανότητά του να συνθέτει συντακτικά έγκυρο κώδικα στη συντριπτική πλειοψηφία των περιπτώσεων. Όσον αφορά την εγκυρότητα του κώδικα τα ποσοστά είναι λίγο χαμηλότερα αγγίζοντας μετά βίας το 50% ωστόσο έχει φανεί πως με μικρές παρεμβάσεις ενός ανθρώπου προγραμματιστή στον παραγόμενο κώδικα αυτά τα ποσοστά μπορούν να προσεγγίσουν ακόμη και το 100%. Ακόμη έχει φανεί πως η αξιοποίηση κατάλληλων επεξηγήσεων του προβλήματος διαθέτει καταλυτικό ρόλο στο επίπεδο εγκυρότητας και ορθότητας του παραγόμενου κώδικα, επιπρόσθετα συνίσταται στον προγραμματιστή, εφόσον είναι δυνατόν, να παραθέτει παραδείγματα του προβλήματος με τις αντίστοιχες επεξηγήσεις τους. Τη σημασία τους φαίνεται να έχουν ακόμη και τα ονόματα τα οποία δίνονται στις συναρτήσεις του προβλήματος στην εκφώνηση καθώς όταν χρησιμοποιούνται ονόματα τα οποία είναι αντιπροσωπευτικά της συνάρτησης το μοντέλο φαίνεται να αντιλαμβάνεται καλύτερα το ρόλο της οδηγώντας σε υψηλότερα ποσοστά ορθότητας [4].

Σε πειράματα που έχουν γίνει σε προβλήματα δύσκολα να επιλυθούν από μεγάλα γλωσσικά μοντέλα, τα αποκαλούμενα big-bench tasks στα οποία τυπικά τα LLMs δεν καταφέρνουν να ξεπεράσουν την ανθρώπινη απόδοση, έχει αναδειχθεί η αποτελεσματικότητα της τεχνικής Chain of thought. Χρησιμοποιώντας την τεχνική CoT, σε έρευνα που πραγματοποιήθηκε, το γλωσσικό μοντέλο Codex κατάφερε να ξεπεράσει την ανθρώπινη απόδοση σε 17 από τα 23 προβλήματα στα οποία δοκιμάστηκε. Η τεχνική CoT γίνεται ακόμα πιο αποτελεσματική σε μοντέλα μεγαλύτερου μεγέθους ενώ τα μικρότερα μοντέλα φαίνεται πως δεν καταφέρνουν να την αξιοποιήσουν τόσο εύκολα. Ακόμη και σε προβλήματα τα οποία τυπικά έχουν μία συγκεκριμένη δυσκολία ασχέτως του μεγέθους του μοντέλου φαίνεται πως η τεχνική Chain of thought καταφέρνει να βελτιώσει την απόδοση του μοντέλου όσο αυξάνει το μέγεθός του [5].

Η παραγωγή prompts τύπου Chain of thought μπορεί να γίνει με δύο διαφορετικές προσεγγίσεις. Η μία είναι η γνωστή, απλή, χειροκίνητη μέθοδος κατά την οποία ο χρήστης εισάγει στο μοντέλο τα βήματα τα οποία επιθυμεί να ακολουθήσει, ενώ η δεύτερη προσέγγιση είναι η λεγόμενη αυτοματοποιημένη μέθοδος. Στην αυτοματοποιημένη μέθοδο ο χρήστης ζητάει από το ίδιο το μοντέλο να σκεφτεί με τη νοοτροπία Chain of thought και να ανακαλύψει μόνο του τα βήματα τα οποία θα πρέπει να ακολουθήσει. Αυτό μπορεί να επιτευχθεί τόσο απλά όσο με την προσθήκη στην αρχή της εισόδου της φράσης "*Let's think step by step*". Η δεύτερη προσέγγιση πέραν της πρακτικής της ευκολίας έχει και το επιπλέον πλεονέκτημα ότι είναι ανεξάρτητη του συγκεκριμένου θέματος ή προβλήματος και μπορεί να γενικευτεί για οποιαδήποτε εργασία. Ωστόσο έχει το μειονέκτημα ότι χωρίς την ανθρώπινη παρέμβαση και προσαρμογή είναι πιθανό να οδηγηθούμε σε αποπροσανατολισμό.

Μία προέκταση της νοοτροπίας Chain of thought η οποία αποσκοπεί στο να δώσει στο μοντέλο τη δυνατότητα να προγραμματίσει τις εργασίες του με τον βέλτιστο τρόπο είναι η μέθοδος Tree of thought. Σε αυτήν τη μέθοδο το μοντέλο δεν παράγει απλώς μία αλυσίδα αλλά ένα ολόκληρο δέντρο από διαφορετικές εναλλακτικές υποαλυσίδες τις οποίες αξιολογεί ώστε να επιλέξει την προτιμότερη από αυτές. Παρόμοια στη λογική σχεδιασμού είναι και η μέθοδος Graph of thought στην οποία, όπως υπαινίσσεται και το όνομα της, το μοντέλο δημιουργεί μόνο του ένα γράφο, ο κάθε κόμβος του οποίου είναι μία εργασία της συνολικής διαδικασίας [6].

Ένα παράδειγμα εφαρμογής της τεχνικής Chain of Thought για την παραγωγή κώδικα για μία ιστοσελίδα θα μπορούσε να είναι το εξής: «*Ακολούθησε τα παρακάτω βήματα για τη δημιουργία μίας ιστοσελίδας:*

1. *Γράψε τον κώδικα HTML για τον σκελετό της σελίδας*
2. *Εμπλούτισε τα γραφικά με CSS*
3. *Πρόσθεσε την λειτουργικότητα με Javascript».*

Έτσι το αποτέλεσμα θα διαθέτει μία λογική δομή αρκετά κοντά σε αυτό που θα έφτιαχνε και μόνος του ο χρήστης και μειώνονται οι πιθανότητες συγγραφής άστοχου κώδικα.

2.1.3 Few shot

Μία άλλη πολλά υποσχόμενη τεχνική είναι αυτή του few shot learning η οποία επιτρέπει σε ένα μοντέλο να μάθει μία νέα διαδικασία με βάση μόνο λίγα παραδείγματα ή με λογικές αναλογίες τις οποίες καλείται να μιμηθεί το μοντέλο. Η προσέγγιση αυτή έρχεται να συμπληρώσει την παραδοσιακή εκπαίδευση των μοντέλων μηχανικής μάθησης η οποία τυπικά χρησιμοποιεί ένα πολύ μεγάλο σύνολο δεδομένων. Το few shot learning αποδεικνύεται ιδιαίτερα χρήσιμο για τα μεγάλα γλωσσικά μοντέλα ιδιαίτερα όσον αφορά εργασίες όπως η μετάφραση, η συγγραφή περιεχομένου, οι απαντήσεις σε ερωτήσεις κ.α. Η εκπαίδευση σε ένα μεγάλο σύνολο δεδομένων μπορεί να είναι πολύ δαπανηρή και απαιτητική σε χρόνο.

Επίσης, η μέθοδος αυτή, μειώνει το κόστος και τον χρόνο εκπαίδευσης των μοντέλων και τους επιτρέπει να μάθουν από νέες πηγές δεδομένων ώστε να είναι πιο εξειδικευμένα, ενημερωμένα και να μπορούν να παράγουν πιο ακριβή αποτελέσματα. Το γεγονός ότι το few shot learning μπορεί να βοηθήσει τα Large Language Models να μάθουν νέες διαδικασίες χρησιμοποιώντας μόνο λίγα παραδείγματα γίνεται χάρη στην ικανότητα αυτών των μοντέλων να εξάγουν χαρακτηριστικά, μοτίβα και πληροφορίες από τα παραδείγματα τα οποία τους δίνονται, να μετασχηματίσουν τα παραδείγματα στη μορφή που τα βολεύει αλλά και να τα χρησιμοποιήσουν ως ενίσχυση στην εμπειρία που ήδη κατέχουν από την εκπαίδευσή τους. Ιδιαίτερα κρίσιμη στην εκπαίδευση μέσω αυτής της τεχνικής είναι η εύρεση των σωστών παραδειγμάτων τα οποία θα πρέπει να είναι επαρκώς αντιπροσωπευτικά της εργασίας χωρίς όμως να γίνονται υπερβολικά εξειδικευμένα. Επίσης η εκμάθηση από λίγα παραδείγματα παρότι είναι ιδιαίτερα πρακτική για τον χρήστη μπορεί να είναι δύσκολη για το μοντέλο καθώς θα πρέπει να είναι σε θέση να γενικεύσει από τα παραδείγματα στις πραγματικές συνθήκες. Ακόμα, η δυνατότητα αξιοποίησης αυτής της τεχνικής μπορεί να μην υπάρχει για ορισμένες εργασίες οι οποίες απαιτούν περισσότερα παραδείγματα.

Ένα παράδειγμα εφαρμογής αυτής της προσέγγισης είναι το εξής: «*Θέλω να μου παραθέσεις πληροφορίες για τα κράτη τις Λατινικής Αμερικής σύμφωνα με το εξής μοτίβο:*

- 1. Ουρουγουάη: Μοντεβιδέο, 3,4 εκ. πληθυσμός*
- 2. Χιλή: Σαντιάγκο, 19,5 εκ. πληθυσμός*

3. *Κολομβία: Μπογκοτά, 51,5 εκ. πληθυσμός κτλ.*».

Έτσι το μοντέλο θα γνωρίζει πως να δομήσει την απάντησή του σύμφωνα με τις οδηγίες μας, όπως την έχουμε στο μυαλό μας.

Μέσα από έρευνες έχει φανεί η μεγάλη βαρύτητα των παραδειγμάτων με τα οποία τροφοδοτούμε τα μεγάλα γλωσσικά μοντέλα. Τα παραδείγματα αυτά θα πρέπει να είναι όσο το δυνατόν σημασιολογικά πλησιέστερα στην εκφώνηση του πραγματικού προβλήματος. Όταν η εκφώνηση των παραδειγμάτων τα οποία παρέχουμε ως είσοδο για υποβοήθηση στο μοντέλο και οι αντίστοιχες απαντήσεις τους ομοιάζουν με το πραγματικό πρόβλημα και με την απάντηση την οποία ζητάμε να λάβουμε, πετυχαίνουμε σημαντικά υψηλότερα ποσοστά επιτυχίας. Ιδίως σε προβλήματα στα οποία επιθυμούμε να λάβουμε μία απάντηση εκτενή σε μέγεθος ή περίπλοκη, το μοντέλο επωφελείται περισσότερο από την μελέτη των παραδειγμάτων [11].

Όσον αφορά τη συγγραφή κώδικα το few shot learning μπορεί να αξιοποιηθεί για την αυτοματοποίηση εργασιών συγγραφής κώδικα όπως η δημιουργία απλών ιστοσελίδων ή η ενσωμάτωση API. Μπορεί επίσης να αξιοποιηθεί για τη δημιουργία νέων μορφών κώδικα όπως η ανάπτυξη νέων τεχνολογιών για την αντιμετώπιση νέων προκλήσεων. Μέσω του few shot learning τα Large Language Models μπορούν να γίνουν πιο ευέλικτα και να προσαρμοστούν σε νέες συνθήκες όπως η συγγραφή κώδικα σε νέες γλώσσες ή σε νέες πλατφόρμες.

Μέσω δοκιμών και ερευνών έχει φανεί πως η χρήση μόνο φυσικής γλώσσας δεν οδηγεί στη δημιουργία ικανοποιητικών αποτελεσμάτων καθώς στερείται καθοδήγησης και ορισμού της μορφής που θα πρέπει να έχουν τα αποτελέσματα. Φαίνεται πως η χρήση αναλυτικών περιγραφών και παραδειγμάτων σχηματισμού του επιθυμητού αποτελέσματος διαδραματίζει σημαντικό ρόλο στην επίλυση του προβλήματος. Σε σειρά πειραμάτων σε διαφορετικά προβλήματα, παρατηρήθηκε ότι τα prompts τα οποία περιείχαν πέρα από φυσική γλώσσα και παραδείγματα οδήγησαν σε συνολικά υψηλότερη επιτυχία αναδεικνύοντας την αξία των παραδειγμάτων [12].

2 Βιβλιογραφική Επισκόπηση – Θεωρητικό Υπόβαθρο

Σε αντίθεση με την κοινή αντίληψη υπάρχουν βάσιμοι ισχυρισμοί πως το few shot learning δεν εισάγει νέα γνώση στο μοντέλο αλλά το βοηθάει να αντιληφθεί καλύτερα το πρόβλημα, να οργανώσει τη γνώση την οποία ήδη κατέχει και να αξιοποιήσει κατάλληλα την εμπειρία του για την επίλυση του συγκεκριμένου προβλήματος. Αξίζει να σημειώσουμε πως η χρήση παραδειγμάτων μπορεί να επιφέρει και αρνητικά αποτελέσματα εάν το μοντέλο αντί να τα αντιληφθεί ως παραδείγματα που αποσκοπούν στο να το καθοδηγήσουν στη συγγραφή αποτελεσμάτων με την κατάλληλη μορφή, τα αντιληφθεί ως τμήμα της επίλυσης του ίδιου του προβλήματος. Αυτό το φαινόμενο αναδεικνύει και τη σημασία της χρήσης κατάλληλων παραδειγμάτων και της παροχής τους με τρόπο ώστε να είναι ξεκάθαρος ο ρόλος τους ως παραδείγματα [13].

Ένα ενδιαφέρον φαινόμενο το οποίο παρατηρείται κατά τη χρήση της τεχνικής few shot είναι η μεταβλητότητα των παραγομένων αποτελεσμάτων η οποία εξαρτάται από τα παραδείγματα τα οποία δίνουμε ως είσοδο στο μοντέλο. Η μεταβλητότητα αυτή μπορεί να οφείλεται σε μία πληθώρα παραγόντων όπως το μέγεθος και η μορφή των παραδειγμάτων ο τρόπος με τον οποίο παρατίθενται ή ακόμα και η σειρά με την οποία παρατίθενται. Ένας από τους βασικούς λόγους είναι αυτό το οποίο αποκαλείται majority level bias, δηλαδή ότι απαντήσεις οι οποίες κυριαρχούν στο prompt τείνουν να κυριαρχούν και στην απάντηση. Για παράδειγμα στην απόπειρα κατηγοριοποίησης ενός κειμένου ως θετικό ή αρνητικό εάν τα περισσότερα παραδείγματα τα οποία έχουν δοθεί στην είσοδο κατηγοριοποιούνται ως θετικά τότε το μοντέλο έχει περισσότερες πιθανότητες να κατηγοριοποιήσει και τα πραγματικά κείμενα ως θετικά. Αυτό το φαινόμενο είναι και ο λόγος για τον οποίο παρατηρούμε συχνά πτώση στην απόδοση των μοντέλων ανάμεσα στο 0 shot και το 1 shot, αφού το παράδειγμα το οποίο δίνεται στην περίπτωση του 1 shot έχει βαρύτερη επίδραση στο αποτέλεσμα στερώντας του την δυνατότητα δημιουργικότητας.

Ένας άλλος λόγος διαφοροποίησης των αποτελεσμάτων είναι το αποκαλούμενο recency bias δηλαδή η τάση του μοντέλου να αντανakλά στις απαντήσεις του τα παραδείγματα τα οποία έχουν δοθεί πιο κοντά στο τέλος του prompt. Αυτό το φαινόμενο έχει ιδιαίτερα μεγάλη ισχύ καθότι η σειρά με την οποία παρατίθενται τα παραδείγματα έχει καθοριστική επιρροή στις απαντήσεις ακόμα και σε περιπτώσεις όπου όλα τα παραδείγματα θα έπρεπε να θεωρούνται ισάξια. Ακόμη όπως είναι λογικό το μοντέλο

τείνει να επαναλαμβάνει πληροφορίες με τις οποίες έχει τροφοδοτηθεί, ενώ δυσκολεύεται να απαντήσει σωστά όταν η απάντηση είναι μία έννοια την οποία έχει συναντήσει μόνο λίγες φορές. Έτσι καταλαβαίνουμε τη σημασία που έχει η σωστή ονομασία των labels κατά την εκπαίδευση [14].

Για παράδειγμα, εάν η είσοδος είναι η εξής: «Θέλω να γράψεις μία ενδεικτική συνάρτηση `rython` σύμφωνα με τα παρακάτω παραδείγματα:

1. `def add(num 1, num 2): etc.`
2. `def greetings(string name): etc.`
3. `def someError(): etc.`».

Είναι πολύ πιθανό η ενδεικτική συνάρτηση που θα παραχθεί εν τέλει να μην διαθέτει παραμέτρους καθότι το τελευταίο παράδειγμα με το οποίο τροφοδοτήσαμε το μοντέλο, δεν διαθέτει παραμέτρους.

Ακόμη είναι σημαντικό να λάβουμε υπόψη μας πως η χρήση υπερβολικά πολλών παραδειγμάτων σε γλωσσικά μοντέλα μικρού μεγέθους, πιθανότατα θα επιφέρει αρνητικά αποτελέσματα στην απόδοση του μοντέλου καθότι λόγω του μικρού του μεγέθους θα δυσκολευτεί να επεξεργαστεί και να αντιληφθεί πλήρως το σύνολο των παραδειγμάτων, οδηγώντας το σε σύγχυση. Έτσι τα μοντέλα με μεγαλύτερο αριθμό παραμέτρων καταφέρνουν να συλλάβουν μεγαλύτερες εισόδους και άρα περισσότερες έννοιες [15].

2.1.4 Άλλες τεχνικές *prompting*

Πέρα από το σύνηθες *prompting* διαλογικής μορφής, υπάρχουν και τα *cloze prompts* τα οποία ζητούν από το μοντέλο να συμπληρώσει το κενό το οποίο υπάρχει σε μία πρόταση. Ενώ τα συνήθη *prompts* είναι καταλληλότερα για δημιουργικές εργασίες, τα *cloze prompts* χρησιμοποιούνται κυρίως σε *masked language models*, δηλαδή μοντέλα τα οποία έχουν εκπαιδευτεί ούτως ώστε να προβλέπουν λέξεις οι οποίες ταιριάζουν στα κενά ενός κειμένου. Ένα παράδειγμα *cloze prompt* είναι το εξής: «Ο ____ είναι ο κορυφαίος αλγόριθμος αναζήτησης.» όπου το μοντέλο καλείται να γεμίσει το κενό *πχ* με την φράση «αλγόριθμος δυαδικής αναζήτησης».

2 Βιβλιογραφική Επισκόπηση – Θεωρητικό Υπόβαθρο

Το prompt ensembling είναι η τεχνική κατά την οποία χρησιμοποιούμε πολλές διαφορετικές απαντήσεις οι οποίες έχουν δοθεί από ένα μοντέλο στο ίδιο πρόβλημα αλλά προέρχονται από εισόδους με ελαφρώς διαφορετική προσέγγιση και διατύπωση. Αντί να επιλέξουμε την καλύτερη από αυτές τις συνδυάζουμε όλες μαζί ώστε να καταλήξουμε σε μία ολοκληρωτική τελική απάντηση. Τα οφέλη του prompt ensembling είναι ότι πρώτον, με αυτόν τον τρόπο μπορούμε να λάβουμε όλα τα πλεονεκτήματα τα οποία μπορεί να βρίσκονται σε κάποια prompts αλλά όχι σε αλλά, δεύτερον, δε χρειάζεται τόσο απαιτητικό prompting καθώς δε μας ενδιαφέρει να καταλήξουμε εξαρχής σε ένα επιτυχημένο prompt αλλά να συνδυάσουμε πολλά λιγότερο επιτυχημένα για την κατασκευή ενός τελικού προϊόντος που θα ικανοποιεί όλες τις απαιτήσεις μας και τρίτον και πολύ σημαντικό είναι ότι με αυτό τον τρόπο μπορεί να σταθεροποιηθεί η απόδοση του μοντέλου σε αλληπάλληλες εργασίες.

Κάποια παραδείγματα διαφορετικών εισόδων για prompt ensembling είναι τα εξής: «Ποιος κέρδισε τη Eurovision το 2005;» και «Ποια χώρα κέρδισε τη Eurovision το 2005;». Εάν οι απαντήσεις είναι οι εξής: «Τη Eurovision του 2005 κέρδισε η Έλενα Παπαρίζου.» και «Το 2005 η χώρα η οποία κατάφερε να επικρατήσει στον διαγωνισμό της Eurovision ήταν η Ελλάδα.», οι απαντήσεις θα μπορούσαν να συνδυαστούν στην πιο πλήρη τελική απάντηση: «Τη Eurovision του 2005 κέρδισε η Ελληνίδα Έλενα Παπαρίζου.»

Το prompt composition είναι μία τεχνική η οποία θυμίζει την λογική του Chain of thought. Στην τεχνική αυτή θεωρούμε ότι έχουμε μία διαδικασία η οποία μπορεί να διαιρεθεί σε υποδιαδικασίες. Έτσι λοιπόν αρχικά διατυπώνουμε τις απαραίτητες οδηγίες για κάθε μία από αυτές τις υποεργασίες και έπειτα τις συνδυάζουμε ώστε να συνθέσουμε ένα συνολικό prompt το οποίο θα καθοδηγεί το μοντέλο ώστε να εκτελέσει τη συνολική διαδικασία [16].

Τεχνικές prompt engineering έχουν χρησιμοποιηθεί ακόμα και για αθέμιτους σκοπούς όπως είναι το jailbreaking του chatGPT με στόχο τη διεξαγωγή παράνομων πράξεων, την εξαπάτηση ή τη συλλογή κατευθύνσεων για διαδικασίες όπως η διανομή κακόβουλου λογισμικού. Τεχνικές όπως η προσποίηση π.χ. της συγγραφής ενός

σεναρίου στο οποίο αναλύεται η διαδικασία διεξαγωγής μιας παράνομης δραστηριότητας, με στόχο την παραπλάνηση του μοντέλου φαίνεται να είναι οι πιο αποδοτικές [17].

2.1.5 Metaprompts

Μία ενδιαφέρουσα τεχνική για τον σχεδιασμό πιο περιγραφικών prompts τα οποία θα δώσουν στο μοντέλο καλύτερη αντίληψη της διαδικασίας την οποία πρέπει να ακολουθήσει είναι η χρήση των αποκαλούμενων metaprompts. Ένα metaprompt είναι στην ουσία μία φράση η οποία εκφράζει μία γενική κατεύθυνση, αυτή η φράση μπορεί να χρησιμοποιηθεί σε συνδυασμό με άλλες πιο εξειδικευμένες πληροφορίες ούτως ώστε να σχηματίσει ένα task specific prompt το οποίο θα καθοδηγήσει το μοντέλο. Αυτή η φράση μπορεί να είναι για παράδειγμα: "*Ας λύσουμε αυτό το πρόβλημα βήμα βήμα*". Τέτοιου είδους φράσεις μπορούν να χρησιμοποιηθούν απαράλλαχτες σε πληθώρα διαφορετικών προβλημάτων προσδίδοντας νόημα και καθοδήγηση στο μοντέλο [13].

2.1.6 Καλές πρακτικές και συστάσεις

Οι πειραματισμοί μας έχουν οδηγήσει σε κάποια ασφαλή συμπεράσματα για τις καλές πρακτικές που θα πρέπει να ακολουθήσουμε κατά τη χρήση των μεγάλων γλωσσικών μοντέλων. Εάν θέλουμε απαντήσεις οι οποίες να είναι ξεκάθαρες και εύστοχες θα πρέπει και εμείς να δώσουμε μια εξίσου ξεκάθαρη περιγραφή του προβλήματος και του ζητούμενου, να διαχωρίσουμε σαφώς τα διαφορετικά τμήματα του προβλήματος ή εάν θέλουμε η απάντηση που θα παραχθεί να είναι ενός συγκεκριμένου μήκους μπορούμε να το διευκρινίσουμε. Ακόμα, είναι πολύ ενδιαφέρουσα η δυνατότητα αυτών των μοντέλων να προσομοιώνουν έναν ειδικό ρόλο και να δώσουν την απάντηση από την οπτική αυτού του ρόλου. Για παράδειγμα θα μπορούσε να ζητηθεί από το μοντέλο να απαντήσει στο ερώτημα: "*Ποια είναι τα οφέλη της πράσινης ενέργειας;*" ως επιστήμονας ή ως πολιτικός ή ως απλός πολίτης. Και μάλιστα για να μειώσουμε την πιθανότητα αναπαραγωγής λάθους πληροφοριών μπορούμε να ζητήσουμε από το μοντέλο να αναζητήσει πληροφορίες σε συγκεκριμένες πηγές και να μας τις παραθέσει. Εδώ αξίζει να σημειώσουμε πως στην περίπτωση του ChatGPT4 έχουν παρατηρηθεί

2 Βιβλιογραφική Επισκόπηση – Θεωρητικό Υπόβαθρο

περιπτώσεις όπου το εργαλείο κάνει αναφορές σε ανύπαρκτες πηγές. Όταν επιθυμούμε να αντιμετωπίσουμε μία απαιτητική εργασία με την προαναφερθείσα τεχνική του διαχωρισμού σε υποεργασίες, είναι σημαντικός ο συντονισμός του εργαλείου σε αλληπάλληλα ερωτήματα.

Για να βοηθήσουμε την αντίληψη του εργαλείου για το πρόβλημα, για την εργασία που έχει ήδη γίνει, τις απαντήσεις και πληροφορίες που έχει ήδη παράγει και για το στάδιο στο οποίο βρισκόμαστε είναι πολύ χρήσιμο σε κάθε νέο ερώτημα να κάνουμε μία σύνοψη και περίληψη των όσων έχουμε ήδη επικοινωνήσει στα προηγούμενα ερωτήματα. Προκειμένου να μειώσουμε την πιθανότητα τυχόν λαθών ή παραλείψεων, μπορούμε να ζητήσουμε από το εργαλείο να ανατρέξει στις προηγούμενες απαντήσεις του και να ελέγξει μήπως έχει ξεχάσει να αναφέρει κάτι ή μήπως έχει αστοχήσει κάπου. Αυτό επιτυγχάνετε με τον εξής απλό τρόπο: «Θα ήθελα να ελέγξεις τις προηγούμενες απαντήσεις σου για τυχόν αστοχίες, λάθη ή παραλείψεις και να με ενημερώσεις εάν εντοπίσεις κάτι αρνητικό». Εάν θέλουμε να ελέγξουμε τις επιδόσεις του μοντέλου αφότου τελειώσουμε την εργασία μας, θα πρέπει να οργανώσουμε τις απαντήσεις που λάβαμε σε μία δομημένη μορφή, έπειτα να ελέγξουμε αν οι απαντήσεις αυτές ικανοποιούν τις απαιτήσεις της διαδικασίας τις οποίες θέσαμε στα ερωτήματα μας και εν τέλει να συγκρίνουμε τις εξόδους του μοντέλου με κάποιες προτυποποιημένες σωστές απαντήσεις στο ίδιο πρόβλημα. Έτσι μπορούμε να ελέγξουμε την ορθότητα, την πληρότητα, την ακρίβεια και τη δομή των απαντήσεων που λάβαμε [18].

Η επιτυχία του prompt engineering εξαρτάται από κάποιες αρχές όπως η σαφήνεια, η ακρίβεια, η ποιότητα της πληροφορίας η οποία παρέχεται στο μοντέλο, η επιθυμητή μορφή του αποτελέσματος και επίσης η πολύ σημαντική αρχή του ελέγχου του μεγέθους της εισόδου αλλά και της εξόδου του προβλήματος. Λέγοντας ποιότητα της πληροφορίας εννοούμε πέρα από την περιγραφή του προβλήματος και τη λογική συνοχή της ίδιας της περιγραφής, δηλαδή εάν τα διάφορα τμήματα, φράσεις και προτάσεις της εισόδου συνδέονται μεταξύ τους με τρόπο ο οποίος έχει νόημα. Πέρα από αυτές τις αυστηρές αρχές υπάρχουν και κάποιες άλλες κατευθύνσεις οι οποίες αν ακολουθηθούν δύνανται να επιφέρουν θετική επίδραση. Αυτές οι κατευθύνσεις περιλαμβάνουν μεταξύ άλλων τη δημιουργικότητα, την εμπειρία, την επιμονή στην επαναληπτική βελτίωση των prompts, το επίπεδο προσαρμογής του prompt στην προσωπικότητα και τις επιθυμίες του

χρήστη κ.α. Βασισμένο σε όλες αυτές τις αρχές και κατευθύνσεις είναι το Clear framework το οποίο αποτελεί έναν οδηγό ο οποίος αποσκοπεί στη μεγιστοποίηση της απόδοσης των μοντέλων τεχνητής νοημοσύνης τύπου GPT. Το Clear framework αποτελείται από πέντε άξονες ο καθένας εκ των οποίων στοχεύει και σε μία ξεχωριστή πτυχή του prompt engineering. Αυτοί οι πέντε άξονες ορίζουν ο καθένας από μία γενική αρχή και κατεύθυνση, σύμφωνα με τις οποίες, το prompt engineering όταν βασίζεται στο συγκεκριμένο framework θα πρέπει να είναι:

- Concise, δηλαδή σύντομο αλλά περιγραφικό και ουσιαστικό, έτσι χωρίς να πλατειάζει και αφαιρώντας φλύαρες πληροφορίες επιτρέπει στο μοντέλο να επικεντρωθεί στα σημαντικότερα σημεία του prompt.
- Logical, δηλαδή οι οδηγίες οι οποίες δίνονται στο μοντέλο να ακολουθούν μία λογική πορεία, να παρέχονται σε μία δομημένη μορφή και να έχουν μεταξύ τους συνοχή ώστε να είμαστε όσο το δυνατόν πιο σίγουροι πως το νόημα το οποίο θέλουμε να περάσουμε θα γίνει κατανοητό και μάλιστα στο σύνολό του.
- Explicit, δηλαδή να ορίζει ξεκάθαρα τους όρους και τις απαιτήσεις του προβλήματος και του χρήστη όπως η μορφή την οποία οφείλει να έχει η επιθυμητή έξοδος και το περιεχόμενό της και πως να περιορίσουμε τις πιθανότητες το αποτέλεσμα να μην ανταποκρίνεται στις προσδοκίες του χρήστη.
- Adaptive, δηλαδή να μην είναι μονολιθικό και μονοδιάστατο αλλά αντιθέτως να έχει αναδειχθεί ως αποτέλεσμα επαναληπτικών πειραματισμών και βελτιώσεων της εισόδου και εξερεύνησης διαφορετικών προσεγγίσεων του προβλήματος μέσω διαφορετικής παράθεσης από τον χρήστη διαφορετικών φράσεων και διαφορετικής δομής του κειμένου εισαγωγής.
- Τέλος, ένας χρήστης ο οποίος ακολουθεί το Clear framework οφείλει να είναι Reflective, δηλαδή οφείλει να αξιολογεί τις απαντήσεις του μοντέλου, να λαμβάνει feedback από άλλους χρήστες, να εντοπίζει περιθώρια βελτίωσης και τυχόν αστοχίες και να επιζητεί την εύρεση των κατάλληλων βελτιώσεων στα σημεία τα οποία δεν είναι ικανοποιητικά [19] [20].

Για να καταλήξουμε στην επιθυμητή απάντηση υπάρχουν κάποιες στρατηγικές prompting τις οποίες μπορούμε να εφαρμόσουμε. Μία έξυπνη στρατηγική είναι το Prompt optimization κατά το οποίο εισάγουμε ένα αρχικό prompt σε ένα γλωσσικό

2 Βιβλιογραφική Επισκόπηση – Θεωρητικό Υπόβαθρο

μοντέλο και του ζητάμε να μας προτείνει βελτιώσεις πάνω σε αυτό οι οποίες είναι πιθανόν να οδηγήσουν σε ένα αποτέλεσμα το οποίο θα προσεγγίζει περισσότερο το επιθυμητό. Εργαλεία όπως το PromptPerfect ή το Promptist, αποσκοπούν στο να βοηθήσουν τους χρήστες να βελτιώσουν τις εισόδους που δίνουν σε μοντέλα τεχνητής νοημοσύνης ασχέτως του ποιο μοντέλο χρησιμοποιούν. Μία άλλη στρατηγική είναι το αποκαλούμενο Recipe library, όπου αξιοποιούμε προϋπάρχοντα templates τα οποία προσαρμόζουμε στην εκάστοτε εργασία [21].

Όταν διαθέτουμε ένα έτοιμο, συγκεκριμένο prompt, μπορούμε να εφαρμόσουμε την αντίστροφη λογική και αντί να προσπαθήσουμε σχηματίσουμε το βέλτιστο prompt, να προσπαθήσουμε να εντοπίσουμε το καταλληλότερο εργαλείο AI. Αυτό μπορεί να γίνει αξιοποιώντας κάποια κριτήρια όπως τη διάρκεια της καθυστέρησης για εργαλεία τα οποία θέτουν ένα μέγιστο μήκος σε αυτήν ή για εργαλεία με μικρό αριθμό παραμέτρων τα οποία ίσως αδυνατούν να αντιληφθούν τα νοήματα ενός μεγάλου prompt στην ολότητά τους, τις επιδόσεις του κάθε μοντέλου στη συγκεκριμένη θεματολογία συγκριτικά με τις επιδόσεις ανταγωνιστικών εργαλείων, τη δυνατότητα του μοντέλου να αντιληφθεί το prompt στην μορφή που βρίσκεται κ.ά. [21]

Ένα διόλου ευκαταφρόνητο ποσοστό χρηστών δηλώνει πως λαμβάνει καλύτερες επιδόσεις από τα Large Language Models όταν τους ζητηθεί να λάβουν το ρόλο ενός ειδήμονα σε ένα αντικείμενο σχετικό με την θεματολογία της συζήτησης, το αποκαλούμενο Role Prompting. Για παράδειγμα μπορούμε στην αρχή του prompt να γράψουμε την φράση "*You are an expert in coding*". Θεωρητικά, η μέθοδος αυτή βοηθάει το μοντέλο ώστε να αντιληφθεί καλύτερα, την προσέγγιση την οποία θα πρέπει να υιοθετήσει, σε ποια σημεία του θέματος πρέπει να εστιάσει, ποια τμήματα από τις γνώσεις του να αξιοποιήσει και πως [22].

Φυσικά πάντα μπορούμε να αξιοποιήσουμε όλα τα υπάρχοντα γλωσσικά μοντέλα ώστε να πάρουμε πολλές διαφορετικές απαντήσεις για την ίδια εργασία και εν τέλει να κρατήσουμε εκείνη η οποία προσεγγίζει περισσότερο τις προτιμήσεις μας [21].

2.1.7 Συχνά λάθη στη συγγραφή prompts

Η συγγραφή επιτυχημένων prompts αποδεικνύεται σύνθετη και ενδιαφέρουσα διαδικασία με αρκετό βάθος και προεκτάσεις. Ο μέσος αρχάριος χρήστης μπορεί να δυσκολευτεί για πολλούς λόγους στη συγγραφή εύστοχων prompts ειδικότερα όταν πρόκειται να χρησιμοποιηθούν αλληπάλλληλα prompts για τη διεκπεραίωση μιας μεγάλης εργασίας. Πολλοί νέοι χρήστες αντιμετωπίζουν την επικοινωνία τους με το μοντέλο όπως την επικοινωνία με έναν άνθρωπο αναμένοντας πως το μοντέλο έχει παρόμοιες ικανότητες και μπορεί να παράγει παρόμοια αποτελέσματα με έναν άνθρωπο. Μερικά από τα συνήθη λάθη των αρχάριων χρηστών είναι πως είτε σταματούν πρόωρα την επικοινωνία τους με το μοντέλο χωρίς να εξετάσουν διαφορετικές προσεγγίσεις στην επίλυση του ίδιου προβλήματος ή εάν οι απαντήσεις που έλαβαν δεν τους ικανοποιούν, σταματούν επίσης πρόωρα χωρίς να εξετάσουν διαφορετικές προσεγγίσεις οι οποίες πιθανόν να οδηγήσουν στην επιθυμητή λύση. Πολλοί από τους νέους χρήστες φαίνεται πως δεν έχουν πλήρη και ξεκάθαρη αντίληψη των δυνατοτήτων ενός μεγάλου γλωσσικού μοντέλου, δε γνωρίζουν πως πρέπει να το χρησιμοποιήσουν, τι απαντήσεις θα πρέπει να περιμένουν κάθε φορά, πως μπορούν να βελτιώσουν τα αποτελέσματα τα οποία λαμβάνουν και ποια είναι τα όρια των δυνατοτήτων του μοντέλου.

Γενικά έχει παρατηρηθεί πολλές φορές το φαινόμενο πως οι χρήστες παρότι παραξενεύονται από τη λειτουργία και τα αποτελέσματα που τους παράγει το μοντέλο τους και ενώ ακόμη μπορεί να πειραματίζονται με τις δυνατότητες του, αδιαφορούν να εξετάσουν συστηματικά την επίδραση που έχουν διαφορετικές είσοδοι και τροποποιήσεις που μπορούν να γίνουν σε αυτές στα διαφορετικά αποτελέσματα τα οποία εν τέλει θα λάβουν. Γενικά οι αρχάριοι χρήστες τείνουν να δείχνουν μία άγνοια και να έχουν αρκετές απορίες για τη συμπεριφορά αυτών των μοντέλων, ωστόσο σπάνια σπεύδουν να διορθώσουν τα prompts τα οποία εισήγαγαν και αντ' αυτού βιάζονται να κάνουν γρήγορες και άστοχες γενικεύσεις για τις τεχνικές prompting που δεν τους ικανοποιούν καθώς και για τις ικανότητες που έχει το κάθε μοντέλο. Γενικότερα ωστόσο φαίνεται πως στις περιπτώσεις όπου οι χρήστες όντως πειραματίστηκαν με διαφορετικές τεχνικές και προσπάθησαν πράγματι να λύσουν προβλήματα τα οποία παρουσιάζονταν

2 Βιβλιογραφική Επισκόπηση – Θεωρητικό Υπόβαθρο

στις απαντήσεις οι οποίες δεν τους ικανοποιούσαν, εν τέλει κατάφεραν να πετύχουν σε μεγάλο βαθμό τους στόχους τους [23].

Όπως έχουμε ήδη αναφέρει για να επιτύχει ένα prompt τον στόχο του με ακρίβεια θα πρέπει να είναι επαρκώς σαφής ο προσδιορισμός του προβλήματος και της επιθυμητής εξόδου. Ως εκ τούτου ένα σύνηθες λάθος στη συγγραφή prompts είναι η ασάφεια, η αοριστία ή αλλιώς η μη επαρκής επεξήγηση του θέματος ή του τύπου της εργασίας που καλείται να εκτελέσει το μοντέλο ή και του ζητούμενου. Στον αντίποδα το overfitting, δηλαδή ο περιορισμός του ζητούμενου σε ένα πολύ στενό πλαίσιο μπορεί επίσης να αποδειχθεί ζημιογόνο καθώς περιορίζει τις δυνατότητες του μοντέλου ως προς τη δημιουργική ευρύτητα. Ένα ακόμη σύνηθες λάθος είναι εισαγωγή προκαταλήψεων από τον χρήστη στο prompt με αποτέλεσμα την αναπαραγωγή στρεβλωμένων αποτελεσμάτων στην έξοδο. Ένα άλλο λάθος το οποίο συχνά περνάει απαρατήρητο και είναι δύσκολο να το αντιληφθούμε εξ αρχής είναι η απαίτηση από το μοντέλο να εκτελέσει εργασίες για τις οποίες δεν έχει εκπαιδευτεί. Υπερεκτιμώντας έτσι τις δυνατότητές του και οδηγώντας το αναγκαστικά στο να μας δώσει άστοχα ή και εντελώς λάθος αποτελέσματα τα οποία δεν ανταποκρίνονται στις απαιτήσεις μας. Για παράδειγμα, εάν ζητήσουμε από ένα μοντέλο να προβλέψει την πορεία της αξίας ενός κεφαλαίου με 100% ακρίβεια προφανώς η απαίτησή μας δε θα ικανοποιηθεί καθώς ακόμη και το ισχυρότερο μοντέλο τεχνητής νοημοσύνης δεν είναι ικανό να φέρει σε πέρας τέτοιου είδους προβλήματα όσο καλώς ορισμένα και αν είναι [24].

Ένα παράδειγμα το οποίο συνοψίζει τις παραπάνω παρατηρήσεις είναι το εξής: «*Θα ήθελα να γράψεις μία συνάρτηση Hello World.*». Το αίτημα αυτό είναι ασαφές καθώς δε διευκρινίζει την επιθυμητή γλώσσα προγραμματισμού, επίσης δε διασαφηνίζεται ρητά εάν θέλουμε να χρησιμοποιηθεί κάποια παράμετρος. Ωστόσο, ο χρήστης δεν πρέπει να απογοητευτεί και να σταματήσει την προσπάθεια εδώ αφού εύκολα μπορεί να αναδιατυπώσει καλύτερα το αίτημα ως: «*Θα ήθελα να γράψεις μία συνάρτηση Hello World στην python. Δε χρειάζονται παράμετροι.*».

2.1.8 Πρότυπα prompting

Τα prompts ανάλογα με τον στόχο, τη μορφή και τις απαιτήσεις τους μπορούν να χωριστούν σε διαφορετικές κατηγορίες μοτίβων. Στο άρθρο [10] παρουσιάζονται τρεις τέτοιες κατηγορίες μοτίβων, μία κατηγορία είναι το Prompt Improvement. Είναι μία πολύ ενδιαφέρουσα και ιδιαίτερη κατηγορία καθώς σε αυτήν ο χρήστης ζητάει από το μοντέλο να του προτείνει διαφορετικά βελτιστοποιημένα τροποποιημένα prompts τα οποία θα μπορούσε ο χρήστης να έχει χρησιμοποιήσει στη θέση κάποιου προηγούμενου. Με αυτόν τον τρόπο ο χρήστης μπορεί να διευκολυνθεί στο να εκφράσει και να δομήσει το ερώτημα του ακόμη και σε τομείς με τους οποίους δεν είναι εξοικειωμένος. Παράδειγμα ενός τέτοιου prompt θα μπορούσε να είναι το εξής: *"Από εδώ και στο εξής για κάθε ερώτημα που θα σου υποβάλλω, πρότεινε μία βελτιωμένη εκδοχή του"*.

Ένα άλλο ιδιαίτερα χρήσιμο πρότυπο μοτίβου είναι το μοτίβο Error Identification κατά το οποίο ο χρήστης ζητάει από το εργαλείο να του εξηγήσει τη λογική πίσω από προηγούμενες απαντήσεις τις οποίες παρήγαγε, για παράδειγμα: *«Μπορείς να μου εξηγήσεις πως κατέληξες στο προηγούμενο αποτέλεσμα;»*. Κατά αυτόν τον τρόπο ο χρήστης μπορεί να ελέγξει την εγκυρότητα των απαντήσεων αλλά και να κατανοήσει τη λογική διαδικασία η οποία οδήγησε σε αυτές. Σε επόμενο στάδιο ο χρήστης κατανοώντας τον τρόπο λειτουργίας τέτοιου είδους μοντέλων, θα είναι σε καλύτερη θέση για να διατυπώσει πιο αποτελεσματικά prompts. Το τρίτο πρότυπο μοτίβου είναι το Interaction στο οποίο ο χρήστης αποσκοπεί στο να δημιουργήσει ένα παιχνίδι το οποίο θα αλληλεπιδρά μαζί του και θα του δίνει τη δυνατότητα να αξιοποιήσει την κριτική του ικανότητα για να επιλύσει προβλήματα πάνω στο σενάριο το οποίο έχει δημιουργήσει.

Στο άρθρο [11] παρουσιάζονται αρκετά ακόμη πρότυπα μοτίβων μεταξύ των οποίων:

- Το Meta Language Creation το οποίο αποσκοπεί στη δημιουργία semantics με το μοντέλο
- Το Fliped Interaction μοτίβο το οποίο είναι πολύ χρήσιμο καθώς δίνει τη δυνατότητα στο εργαλείο τεχνητής νοημοσύνης να κάνει ερωτήσεις στον χρήστη για διευκρινήσεις και για την απόκτηση επιπλέον πληροφοριών
- Το μοτίβο Persona κατά το οποίο το μοντέλο υποδύεται έναν ρόλο και δίνει απαντήσεις με βάση την οπτική αυτού του ρόλου

2 Βιβλιογραφική Επισκόπηση – Θεωρητικό Υπόβαθρο

- Το Fact Checklist μοτίβο το οποίο θέτει το μοντέλο στη διαδικασία ελέγχου μίας λίστας απαιτήσεων οι οποίες θα πρέπει να ικανοποιούνται στην παραγόμενη έξοδο
- Το Template μοτίβο το οποίο καθορίζει μία σαφή δομή για τις εξόδους
- Το Infinite Generation μοτίβο το οποίο αναπαράγει αλληπάλλληλες απαντήσεις αυτόματα
- Το Visualization Generator μοτίβο το οποίο χρησιμοποιεί κείμενο για να δημιουργήσει εικόνες
- Το Context Manager μοτίβο, μία επίσης πολύ ενδιαφέρουσα ιδέα η οποία επιτρέπει στον χρήστη να αφαιρέσει συγκεκριμένα τμήματα από τη συζήτηση με το εργαλείο βοηθώντας το να εστιάσει στις πιο σημαντικές πληροφορίες χωρίς να παραπλανάται από διάφορες πληροφορίες που αναφέρθηκαν κατά την συνομιλία κ.ά. [25] [26].

2.1.9 Τεχνικές prompting για τη συγγραφή κώδικα

Η επιτυχία των μεγάλων γλωσσικών μοντέλων τα οποία έχουν επιδείξει εκπληκτικές ικανότητες σε ένα εύρος προγραμματικών εργασιών μετατρέποντας εισόδους φυσικής γλώσσας σε αντίστοιχα προγράμματα κώδικα έχει φτάσει σε σημείο αυτού του είδους τα εργαλεία να ανταγωνίζονται πλέον τους ανθρώπους προγραμματιστές [27].

Μάλιστα σε έρευνες οι οποίες συγκρίνουν την ικανότητα του εργαλείου Copilot να γράφει κώδικα με τη συγγραφή κώδικα από ανθρώπους προγραμματιστές και πιο συγκεκριμένα με την τεχνική Pair Programming φαίνεται πως το Copilot καταφέρνει να γράφει περισσότερο κώδικα σε μικρότερο χρονικό διάστημα από τους ανθρώπους, ωστόσο ο κώδικας αυτός συνήθως δε βρίσκεται στο ίδιο ποιοτικό επίπεδο με αυτό το οποίο πετυχαίνουν οι άνθρωποι προγραμματιστές [28].

Έχουν προταθεί πολλά διαφορετικά μοτίβα χρήσης των Large Language Models για τη συγγραφή κώδικα ανάλογα με τον στόχο τον οποίο θέλει να πετύχει ο προγραμματιστής και μπορούν να χωριστούν σε δύο βασικές κατηγορίες: η μία

εμπεριέχει τα μοτίβα τα οποία αφορούν τον έλεγχο της ποιότητας και τις τροποποιήσεις σε αυτόν και η άλλη τις απαιτήσεις του συστήματος και την αρχιτεκτονική. Το κάθε μοτίβο διαφοροποιείται ανάλογα με τον στόχο τον οποίο καλείται να εξυπηρετήσει. Παραδείγματος χάριν θα χρησιμοποιηθούν διαφορετικές προσεγγίσεις για την επεξήγηση του κώδικα από ένα LLM, για το refactoring και την αναδιοργάνωση του κώδικα, για την προσομοίωση της εκτέλεσης ενός προγράμματος, για την τροποποίηση του κώδικα σε περίπτωση μεταβολής των απαιτήσεων του προγραμματιστή, για τον πειραματισμό με διαφορετικές αρχιτεκτονικές κ.ά. Ακόμη ξεχωρίζει το πρότυπο μοτίβου το οποίο αποσκοπεί στη δημιουργία παραδειγμάτων εκτέλεσης του κώδικα τα οποία αργότερα θα ανατροφοδοτηθούν σε αυτόν ως μέρος της τεχνικής few shot prompting. Τα μοτίβα αυτά ουσιαστικά αποτελούν διαφορετικά πρότυπα διατύπωσης όπου φαίνεται πως όταν αλλάζει το ζητούμενο του προβλήματος, η δομή αλλά και οι τεχνικές που χρησιμοποιούνται για τη διατύπωση του prompt είναι εντελώς διαφορετικά [29].

Η επιτυχία της συμβολής του prompt engineering στα γλωσσικά μοντέλα κώδικα μπορεί να αξιολογηθεί με βάση το αν έλυσαν το πρόβλημα το οποίο η αρχική είσοδος στο μοντέλο είχε αποτύχει να λύσει. Τέτοιου είδους πειράματα και αξιολόγηση έχουν γίνει στο μεγάλο γλωσσικό μοντέλο κώδικα Copilot υπό την αιγίδα του Github. Το μοντέλο φαίνεται πως αρχικά είχε μία μέτρια απόδοση καταφέροντας να λύσει σχεδόν τα μισά προβλήματα με την αρχική τους διατύπωση. Ωστόσο η τροποποίηση της περιγραφής αυτών των προβλημάτων ήταν αρκετή ώστε το μοντέλο να καταφέρει να λύσει τα περισσότερα από τα προβλήματα τα οποία αρχικά το δυσκόλεψαν. Ιδιαίτερο ενδιαφέρον παρουσιάζει το γεγονός πως το μοντέλο φαίνεται να ανταποκρίνεται καλύτερα σε οδηγίες οι οποίες ομοιάζουν δομικά με ψευδοκώδικα. Ακόμα και έτσι, τα προβλήματα τα οποία δυσκόλεψαν περισσότερο το μοντέλο ήταν αυτά τα οποία είχαν τις πιο μακροσκελείς και φλύαρες περιγραφές ακόμα και όταν αυτές ήταν δομημένες με μορφή ψευδοκώδικα. Το γεγονός αυτό έχει ιδιαίτερο ενδιαφέρον καθώς δείχνει πως παρότι συνήθως οι αναλυτικές περιγραφές του ζητούμενου βοηθούν το μοντέλο να το αντιληφθεί σε μεγαλύτερο βάθος και να παράγει αντιστοίχως πιο εύστοχα αποτελέσματα, υπάρχουν και περιπτώσεις όπου οι υπερβολικά φλύαρες περιγραφές με πολλά λόγια ίσως είναι δύσκολο να γίνουν αντιληπτές στην ολότητά τους από το μοντέλο προκαλώντας του σύγχυση [30].

2 Βιβλιογραφική Επισκόπηση – Θεωρητικό Υπόβαθρο

Έπειτα από πειραματισμούς και έρευνες, έχει φανεί ότι όπως σε κάθε άλλη εργασία έτσι και στη συγγραφή κώδικα ο σχεδιασμός της εισόδου η οποία θα δοθεί στο μοντέλο έχει καταλυτική σημασία για την απόδοση του, αυτό ισχύει τόσο για τις εργασίες τύπου text to code όσο και για τις εργασίες τύπου code to code. Τα πειράματα έδειξαν πως πολλές φορές ο κώδικας ο οποίος παράγεται από το μοντέλο αν και συνήθως συντακτικά σωστός, απέχει από το βέλτιστο κυρίως καθότι είναι εκτενέστερος από όσο χρειάζεται. Για να επιλυθεί αυτό το πρόβλημα είναι απλώς αρκετό να ζητήσουμε από το μοντέλο να παράγει συνοπτικότερες απαντήσεις. Σημειώνεται ωστόσο πως παρότι στην παραγωγή κώδικα από φυσική γλώσσα το αίτημά μας για συνοπτικότερες απαντήσεις τυπικά έχει θετικά αποτελέσματα, δεν ισχύει το ίδιο για τα code to code tasks όπου εκεί επιφέρει ελαφρώς αρνητικά αποτελέσματα. Όπως ήδη γνωρίζουμε μοντέλα όπως το chatGPT και το Bard, χρησιμοποιούν sessions αλληλεπιδράσεων με τον κάθε χρήστη το οποίο διατηρεί το δικό του ιστορικό και τα μοντέλα αυτά αξιοποιούν το ιστορικό τους προκειμένου να αντιληφθούν τη θεματολογία της συζήτησης και να παράγουν τις σχετικότερες απαντήσεις. Το ίδιο ισχύει και για τη συγγραφή κώδικα όπου αξιοποιώντας τα sessions, τα μοντέλα αντιλαμβάνονται καλύτερα το αντικείμενο του αιτήματός μας και παράγουν πιο εύστοχες απαντήσεις.

Ακόμα γνωρίζουμε ότι το chatGPT διαθέτει ένα βαθμό τυχαιότητας, δηλαδή εάν του ζητήσουμε επανειλημμένα να μας γράψει κώδικα για το ίδιο πρόβλημα μπορεί κάθε φορά ο κώδικας να έχει ελαφρές διαφοροποιήσεις ώστε να προάγεται η δημιουργικότητα. Αυτό δεν αποδεικνύεται αρκετά σημαντικό για να επηρεάσει αρνητικά την αποτελεσματικότητα των απαντήσεων [31].

Όπως και στις περισσότερες εργασίες έτσι και στη συγγραφή κώδικα, μέσα από πειραματισμούς καταλήγουμε στο συμπέρασμα πως ο συνδυασμός χρήσης φυσικής γλώσσας και ενδεικτικών τμημάτων κώδικα για την περιγραφή των απαιτήσεων του κώδικα και του προβλήματος το οποίο θα πρέπει να λυθεί, σε σύγκριση με τη χρήση μόνο φυσικής γλώσσας ή μόνο κώδικα, αυξάνει ακόμα περισσότερο τις πιθανότητες να παραχθούν σωστές απαντήσεις οι οποίες επιλύουν το πρόβλημα. Επίσης, η παράθεση κατάλληλων παραδειγμάτων έχει καταλυτική επίδραση στη συμπεριφορά του μοντέλου βοηθώντας το να γράψει πιο επιτυχημένο κώδικα. Πέρα από τις τεχνικές prompt

engineering, όπως είναι αναμενόμενο, τα μεγαλύτερα σε μέγεθος νευρωνικά δίκτυα διαθέτουν υψηλότερες ικανότητες παραγωγής επιτυχημένου κώδικα ιδιαιτέρως στα πιο δύσκολα προβλήματα [12].

Τα ονόματα τα οποία δίνουμε στις συναρτήσεις μας πέραν του ότι βοηθάνε τους ανθρώπους να αντιληφθούν τη λογική του κώδικα φαίνεται πως ενισχύουν και την αντίληψη των μεγάλων γλωσσικών μοντέλων. Για παράδειγμα, είναι προτιμότερο να ονομάσουμε μία συνάρτηση ταξινόμησης ‘`sortingFunction`’ παρά ‘`functionA`’ καθότι το πρώτο όνομα είναι πολύ πιο επεξηγηματικό βοηθώντας όχι μόνο τους ανθρώπους αλλά και τα εργαλεία τεχνητής νοημοσύνης. Έχει παρατηρηθεί πως αντικαθιστώντας τα επεξηγηματικά, χαρακτηριστικά ονόματα συναρτήσεων με άσχετες και τυχαίες λέξεις, η εγκυρότητα του παραγόμενου κώδικα μειώνεται. Η έρευνα στο [32] επιβεβαιώνει για άλλη μία φορά την παρατήρηση ότι για να πετύχουμε την υψηλότερη δυνατή απόδοση κατά την συγγραφή κώδικα χρειαζόμαστε και prompts με φυσική γλώσσα για την περιγραφή του προγράμματος που θέλουμε να παραχθεί και των συναρτήσεων και των παραμέτρων που αυτές θέλουμε να έχουν αλλά και ενδεικτικά τμήματα κώδικα στα οποία θα φαίνονται τα ονόματα και οι παράμετροι των συναρτήσεων. Μάλιστα συστήνεται οι περιγραφές να είναι όσο το δυνατόν ακριβέστερες και πιο ξεκάθαρες [32].

Παρόμοια συμπεράσματα έχουν βρεθεί και για άλλα εργαλεία συγγραφής κώδικα όπως το Alpha code όπου φαίνεται πως απλούστερες αλλά ταυτόχρονα επαρκείς περιγραφές του ίδιου προβλήματος βελτιώνουν την απόδοση του μοντέλου επιφέροντας υψηλότερα ποσοστά επιτυχημένων λύσεων [33]. Για παράδειγμα, το «*Γράψε μια συνάρτηση Hello World στην python, χωρίς παραμέτρους.*» είναι προτιμότερη είσοδος από το «*Γράψε μια συνάρτηση η οποία θα εμφανίζει στην οθόνη την φράση ‘Hello World’, ονόμασέ την Hello World, γράψε την στη γλώσσα προγραμματισμού python και θυμήσου πως τόσο απλές συναρτήσεις δε χρειάζονται παραμέτρους*», αφού και τα δύο, επί της ουσίας, εμπεριέχουν την ίδια πληροφορία αλλά το πρώτο είναι πιο λακωνικό.

Παρόμοια αποτελέσματα όσον αφορά τον εμπλουτισμό της εισόδου με `query plans` και την παροχή παραδειγμάτων έχουν παρατηρηθεί και στη μηχανή `Codex db` για την επεξεργασία `sql queries`. Σημειώνεται ωστόσο πως τα υπερβολικά μακροσκελή

2 Βιβλιογραφική Επισκόπηση – Θεωρητικό Υπόβαθρο

prompts, όπως σε περιπτώσεις όπου αυτό οφείλεται στην παροχή υπερβολικά πολλών παραδειγμάτων, δε βοηθάνε το μοντέλο στη διεκπεραίωση της εργασίας του καθώς ξεπερνάνε το μέγιστο όγκο της δυνατής αντίληψης που διαθέτει το μοντέλο [34].

Σε εργασίες συγγραφής κώδικα είναι απολύτως απαραίτητο να περιγράψουμε με ακρίβεια, σαφήνεια και με κατανοητό τρόπο το ζητούμενο το οποίο επιθυμούμε να παραχθεί. Αυτό σημαίνει ότι απαιτούνται καλώς ορισμένες συναρτήσεις με αναλυτική αναφορά των μεταβλητών που χρησιμοποιούν και των λειτουργιών που επιτελούν. Πολλά προβλήματα στον παραγόμενο κώδικα προκαλούνται από ασάφειες, ελλείψεις, αστοχίες και λογικά λάθη στα prompts και μπορούν εύκολα να διορθωθούν αναδιατυπώνοντας ορθότερα το ζητούμενο.

Ενδιαφέρον προκαλεί το γεγονός ότι το Codex έχει παρατηρηθεί να παράγει μαζί με τον κώδικα και ακατάλληλα σχόλια τα οποία χαρακτηρίζονται από προκαταλήψεις ή άλλες αθέμιτες αντιλήψεις όπως π.χ. ο ρατσισμός. Ακόμη έχει παρατηρηθεί ειδικά για το εργαλείο Codex ότι ο τρόπος με τον οποίο διαχειρίζεται τις εισόδους του χρήστη είναι ότι προσπαθεί να εναρμονιστεί με το ύφος τους και να δημιουργήσει απαντήσεις οι οποίες συνεχίζουν τη νοοτροπία της εισόδου. Αυτό σημαίνει πως prompts διατυπωμένα με μπερδεμένα λόγια θα οδηγήσουν και σε μπερδεμένες απαντήσεις, prompts με προκαταλήψεις οδηγήσουν σε απαντήσεις με προκαταλήψεις, ασαφείς είσοδοι θα οδηγήσουν σε ασαφείς εξόδους κ.τ.λ. Μάλιστα τα ικανότερα μοντέλα με περισσότερες παραμέτρους, φαίνεται να επηρεάζονται σε μεγαλύτερο βαθμό από αστοχίες και λάθη στις εισόδους τους κάτι το οποίο εξηγείται ως απόρροια της μεθόδου με την οποία λειτουργούν σύμφωνα με την οποία η παραγόμενη απάντηση θα πρέπει να ακολουθεί τη λογική της εισόδου. Έτσι τα πιο πολύπλοκα μοντέλα έχουν την τάση να ακολουθούν πιο πιστά ακόμα και τις κακές εισόδους τους επιφέροντας αναλόγως κακές εξόδους [33].

Μία άλλη έρευνα η οποία πειραματίστηκε με την συγγραφή κώδικα OCL για UML με τη χρήση του μοντέλου Codex, κατέληξε σε παρόμοια συμπεράσματα. Φαίνεται πως τα prompts τα οποία χρησιμοποιούσαν μόνο φυσική γλώσσα για την περιγραφή των OCL constraints, δυσκόλευαν το μοντέλο να παράγει τις επιθυμητές απαντήσεις, και οι απαντήσεις τις οποίες παρήγαγε παρότι δεν απείχαν υπερβολικά από τον κώδικα τον οποίο θα έγραφε ένας άνθρωπος, δεν ήταν τόσο εύστοχες όσο οι απαντήσεις οι οποίες

βασίζονταν σε prompts τα οποία περιείχαν και κάποια ενδεικτικά τμήματα UML. Τα prompts τα οποία περιλάμβαναν τμήματα UML, οδήγησαν σε σημαντικά υψηλότερα ποσοστά επιτυχίας και σε κώδικα ο οποίος ήταν πιο κοντά στον ιδανικό, όπως θα τον έγραφε ένας έμπειρος προγραμματιστής. Στη συνέχεια οι ερευνητές πρόσθεσαν στις εισόδους πέρα από τμήματα UML και κάποια παραδείγματα τύπου few shot. Τα ευρήματα τους δείχνουν πως η χρήση κατάλληλων παραδειγμάτων έχει τη δυνατότητα να βελτιώσει ακόμα περισσότερο την αξιοπιστία των απαντήσεων οι οποίες μάλιστα είναι ακόμη πιο κοντά στα ιδανικά UML constraints τα οποία θα έγραφε ένας άνθρωπος.

Συμπερασματικά δηλαδή, η χρήση ενδεικτικών τμημάτων κώδικα μέσα στα prompts βελτιώνει τις απαντήσεις συγκριτικά με τη χρήση μόνο φυσικής γλώσσας, ενώ η προσθήκη εύστοχων παραδειγμάτων βοηθάει ακόμα περισσότερο το μοντέλο να προσεγγίσει τις βέλτιστες δυνατές απαντήσεις [35]. Για παράδειγμα, αντί να γράψουμε «Εξήγησέ μου τι κάνει ο πρώτος βρόγχος του προγράμματος.» είναι καλύτερο να γράψουμε «Τι κάνει η γραμμή `for x in array:`».

Σε μία έρευνα η οποία ανέλυσε τη συγγραφή κώδικα και την επεξεργασία queries μέσω του εργαλείου Codex αναδεικνύεται για άλλη μία φορά η αποτελεσματικότητα της τεχνικής few shot. Ακόμα για άλλη μία φορά βλέπουμε πως και σε αυτό το είδος συγγραφής κώδικα ο συνδυασμός φυσικής γλώσσας και κώδικα, στην προκειμένη περίπτωση queries, στην είσοδο του εργαλείου, υπερέχει σημαντικά από τη χρήση είτε μόνο φυσικής γλώσσας είτε μόνο κώδικα, με την απόδοση να μεταβάλλεται αισθητά όσο διαφοροποιούνται τα prompts. Στα πειράματα που πραγματοποιήθηκαν στο μοντέλο Davinci, φάνηκε πως η χρήση δύο παραδειγμάτων ανά prompt, σε αντίθεση με τα 0 και 4, οδήγησε στο να λυθεί ο μέγιστος αριθμός queries. Παράλληλα, σε άλλα πειράματα έχει φανεί πως η χρήση ενός και μόνο παραδείγματος σε πολλές περιπτώσεις όχι μόνο δε βοηθάει το εργαλείο να καταλάβει αλλά μπορεί ακόμα και να μειώσει την απόδοσή του αφού του στερεί τη δυνατότητα δημιουργικότητας. Επιπλέον, είναι λογικό και ξεκάθαρο στα πειράματα πως η διαδικασία fine tuning του μοντέλου συγγραφής κώδικα δύναται να αυξήσει την ακρίβεια του μοντέλου χρησιμοποιώντας ταυτόχρονα εισόδους μικρότερου μεγέθους [36].

2 Βιβλιογραφική Επισκόπηση – Θεωρητικό Υπόβαθρο

Σε μία σειρά πειραμάτων όπου ζητήθηκε από προγραμματιστές να χρησιμοποιήσουν το εργαλείο Copilot για τη συγγραφή κώδικα, οι συμμετέχοντες βρήκαν το Copilot ιδιαίτερα χρήσιμο καθώς τους παρείχε ένα starting point προγράμματος ενώ εάν το έγραφαν μόνοι τους εξαρχής θα ξεκινούσαν από το μηδέν. Ωστόσο οι συμμετέχοντες εντόπισαν τρία εμπόδια στην ολοκλήρωση των πειραμάτων. Το πρώτο ήταν πως το Copilot συχνά αποτύγχανε να εντοπίσει και να αντιμετωπίσει τα σφάλματα τα οποία το ίδιο παρήγαγε στον κώδικα. Το δεύτερο, ως συνέπεια του πρώτου, ήταν η υποτίμηση του κόπου που χρειάστηκε για τη διόρθωση αυτών των σφαλμάτων. Για τη διόρθωσή τους οι συμμετέχοντες ως επί το πλείστον επέλεξαν ανάμεσα σε δύο προσεγγίσεις η μία ήταν να επιχειρήσουν να διορθώσουν το πρόβλημα στον κώδικα ο οποίος παρήχθη και η δεύτερη χρησιμοποιήθηκε σε περιπτώσεις όπου οι συμμετέχοντες δεν ήταν πρόθυμοι να διορθώσουν το πρόβλημα οπότε απέρριψαν εντελώς τον κώδικα ο οποίος παρήχθη και προσπάθησαν να εντοπίσουν καλύτερες λύσεις στο διαδίκτυο.

Το τρίτο εμπόδιο το οποίο εντόπισαν ήταν η ευαισθησία του Copilot στα σχόλια των προγραμματιστών τα οποία αποσκοπούσαν στο να το κατευθύνουν στην εργασία του με αποτέλεσμα. Ακόμη και μικρές τροποποιήσεις στα σχόλια οδηγούσαν σε σημαντικές διαφοροποιήσεις στο αποτέλεσμα. Ανάμεσα στους χρήστες οι οποίοι κατάφεραν να λύσουν ακόμη και το πιο δύσκολο πρόβλημα το οποίο τους ανατέθηκε υπήρχαν κάποιοι οι οποίοι το πέτυχαν διαιρώντας το σε υπό προβλήματα και παρέχοντας στο Copilot ξεχωριστές οδηγίες επίλυσης του κάθε προβλήματος. Με αυτόν τον τρόπο πέτυχαν υψηλότερη αποδοτικότητα και καλύτερο user experience. Παρόλα αυτά έχει σημασία να σχολιαστεί πως σε αντίθεση με άλλες έρευνες, στη συγκεκριμένη δε φάνηκαν να υπάρχουν σημαντικά κέρδη στη χρονική διάρκεια της επίλυσης των προβλημάτων με τη βοήθεια του Copilot συγκριτικά με την επίλυση τους μόνο από ανθρώπους χωρίς βοήθεια. Οι συμμετέχοντες όμως προτίμησαν τη χρήση του εργαλείου καθότι η συνδρομή του κατέστησε την όλη διαδικασία ευκολότερη [37].

Μία ενδιαφέρουσα έρευνα ασχολείται με πειράματα τα οποία αποσκοπούν στο να αξιολογήσουν την ικανότητα του εργαλείου Copilot στην αξιολόγηση της πολυπλοκότητας προγραμμάτων κώδικα. Η έρευνα καταλήγει πως παρά την ύπαρξη εργαλείου το οποίο επιτελεί την εκτίμησή της πολυπλοκότητας, το εργαλείο αυτό δεν αποδεικνύεται πάντα ικανό ώστε να αντιληφθεί την πολυπλοκότητα της εκτέλεσης του

κώδικα. Άλλα εργαλεία deep learning τα οποία εξειδικεύονται στην εκτίμηση πολυπλοκότητας καταφέρνουν να ξεπεράσουν τις ικανότητες του Copilot στο συγκεκριμένο πρόβλημα [38].

Υπάρχει μία τεκμηριωμένη θεωρία σύμφωνα με την οποία τα μεγάλα γλωσσικά μοντέλα τα οποία είναι εκπαιδευμένα σε κώδικα αντιλαμβάνονται πέραν των άλλων, με μεγάλη επάρκεια, δομημένες λογικές έννοιες όπως για παράδειγμα καλώς ορισμένες διαδικασίες και λογικά γραφήματα. Είναι δηλαδή εξαιρετικά ικανά στο να αναπαράγουν με δομημένο τρόπο, όπως είναι ο κώδικας, στα αποτελέσματά τους, λογικές έννοιες τις οποίες έχουν λάβει ως είσοδο. Για να επιβεβαιώσουν αυτή τη θεωρία έχουν γίνει πειράματα τα οποία χρησιμοποιούν το γλωσσικό μοντέλο κώδικα Codex της OpenAI σε σύγκριση με γλωσσικά μοντέλα φυσικής γλώσσας. Στα πειράματα αυτά ζητείται από τα μοντέλα να αναπαράγουν με δομημένο τρόπο λογικές έννοιες όπως καλώς ορισμένες διαδικασίες όπου το Codex δείχνει να υπερέχει, παρουσιάζοντας πιο ικανοποιητικές αποδόσεις στο συγκεκριμένο είδος προβλήματος [39].

Μία ενδιαφέρουσα έρευνα εξετάζει τη δυνατότητα των Large Language Models στη διενέργεια Event Argument Extraction, δηλαδή να μετατρέπουν αδόμητες πληροφορίες οι οποίες τους παρέχονται σε φυσική γλώσσα, σε δομημένο κώδικα. Για να το πετύχουν αυτό οι ερευνητές χρησιμοποιούν το εργαλείο Code4Strukt, στο εργαλείο αυτό παρέχουν εισόδους οι οποίες αποτελούνται από τρία τμήματα. Στο πρώτο τμήμα παρουσιάζεται η δομή του κώδικα ο οποίος θα πρέπει να παραχθεί, δηλαδή παρουσιάζονται οι κλάσεις και οντότητες οι οποίες θα πρέπει να εντοπιστούν και να χαρτογραφηθούν στην απάντηση. Στο δεύτερο μέρος μπορούμε, εάν το επιθυμούμε, να παρέχουμε κάποια παραδείγματα σύμφωνα με τη γνωστή τεχνική few shot, για το γεγονός το οποίο περιγράφεται στο πρώτο τμήμα.

Τέλος, στο τρίτο μέρος της εισόδου, παρέχουμε τις οδηγίες για την εργασία η οποία επιθυμούμε να εκτελεστεί. Κατά τη διαδικασία αυτήν παρατηρήθηκε ότι η χρήση φυσικής γλώσσας σε σημεία του prompt στα οποία θα μπορούσε να χρησιμοποιηθεί κώδικας, όπως για παράδειγμα στο πρώτο τμήμα, επέφερε σημαντικά μειωμένη αποτελεσματικότητα σε όλα τα μοντέλα τα οποία συμμετείχαν στην έρευνα. Όσον

2 Βιβλιογραφική Επισκόπηση – Θεωρητικό Υπόβαθρο

αφορά το δεύτερο τμήμα, τα παραδείγματα τα οποία παρέχονται προαιρετικά προκαλούν μία φθίνουσα αύξηση της αποτελεσματικότητας με το πρώτο παράδειγμα να τους φέρει το υψηλότερο κέρδος και το τελευταίο παράδειγμα να προσφέρει πάλι κέρδος, όχι όμως στον ίδιο βαθμό. Αυτό εξηγείται ως αποτέλεσμα του κορεσμού των παραδειγμάτων αφού από ένα σημείο και μετά οι περισσότερες εναλλακτικές έχουν ήδη παρουσιαστεί [40].

Σε ένα πείραμα όπου ζητήθηκε από το chatGPT να παράγει πολλές διαφορετικές λύσεις για το ίδιο προγραμματιστικό πρόβλημα και να επιλέξει την προτιμότερη από αυτές, φάνηκε πως καταφέρνει να επιλέγει όντως τις καλύτερες απαντήσεις οι οποίες μάλιστα ανταγωνίζονται και ξεπερνούν τις καλύτερες απαντήσεις οι οποίες δόθηκαν από ανθρώπους [41].

Στο [42] παρουσιάζεται μία πολύ ενδιαφέρουσα μέθοδος κατά την οποία ζητείται από το μεγάλο γλωσσικό μοντέλο παράγει μόνο του παραδείγματα λυμένων προβλημάτων αντίστοιχα με το πραγματικό πρόβλημα το οποίο του ζητάμε να απαντήσει. Η μέθοδος αυτή αποσκοπεί στο να μειώσει τον χρόνο και τον κόπο που απαιτείται για την παροχή labeled παραδειγμάτων στο μοντέλο. Η μέθοδος αντλεί έμπνευση από τον τρόπο με τον οποίο οι άνθρωποι προσεγγίζουν την επίλυση νέων προβλημάτων όταν τα συναντάνε, ο τρόπος αυτός είναι ότι οι άνθρωποι αξιοποιούν τις γνώσεις και την εμπειρία την οποία έχουν συλλέξει στο παρελθόν μέσα από παρόμοια προβλήματα τα οποία κλήθηκαν να αντιμετωπίσουν ή από την εκπαίδευσή τους. Οι άνθρωποι χρησιμοποιούν υπό μία έννοια την μέθοδο Few shot για την λύση των προβλημάτων τους διαθέτοντας όμως ήδη έτοιμα labeled παραδείγματα από τις εμπειρίες της ζωής τους.

Παράλληλα, η μέθοδος αυτή, διαθέτει άλλη μία πρακτική ευκολία καθώς τα παραδείγματα τα οποία παράγει το μοντέλο αυτομάτως για τον εαυτό του είναι εξειδικευμένα σχετικά με το πραγματικό πρόβλημα. Επιπρόσθετα μαζί με τη δημιουργία σχετικών παραδειγμάτων μπορούμε να ζητήσουμε από το εργαλείο να δημιουργήσει ένα είδος tutorial με οδηγίες για την επίλυση του προβλήματος, τις οποίες θα ακολουθήσει και το ίδιο. Οι πειραματισμοί με αυτήν την μέθοδο έδειξαν πως καταφέρνει να ξεπεράσει σε επιδόσεις την απλή τεχνική Chain of thought χωρίς παραδείγματα και την

παραδοσιακή τεχνική Few shot CoT στα μοντέλα GPT3.5 και GPT4, στην παραγωγή κώδικα, ενώ ξεπερνάει την μέθοδο της CoT και στα αποκαλούμενα BIG bench tasks [42].

Υπάρχουν εξειδικευμένα μοτίβα prompting για τη συγγραφή κώδικα. Ένα από αυτά είναι το αποκαλούμενο Code Clustering Pattern κατά το οποίο ζητάμε από το εργαλείο τεχνητής νοημοσύνης να δημιουργήσει κώδικα ο οποίος είναι οργανωμένος σε συναρτήσεις, κλάσεις κ.τ.λ. Έτσι ώστε να είμαστε σίγουροι ότι ο κώδικας που θα παραχθεί θα έχει μία οργανωμένη δομή. Ένα άλλο μοτίβο είναι το αποκαλούμενο Intermediate Abstraction Pattern, αυτό το μοτίβο είναι πολύ σημαντικό καθώς ζητά από το εργαλείο να καθιστά εμφανή την επιχειρηματική λογική του παραγόμενου κώδικα από τυχόν βιβλιοθήκες οι οποίες χρησιμοποιούνται, ούτως ώστε η λογική του κώδικα να είναι εμφανής και να είναι δυνατή η αντικατάσταση των βιβλιοθηκών. Ένα ακόμα pattern, ίσως το απλούστερο αλλά και το βασικότερο όλων, είναι το αποκαλούμενο Principled Code Pattern το οποίο απλώς ζητεί από το εργαλείο να τηρήσει όλες τις τυπικές καλές πρακτικές συγγραφής κώδικα. [32] [72]

2.1.10 Αποσφαλμάτωση μέσω LLMs

Στο [24] εξετάζονται οι δυνατότητες των μεγάλων γλωσσικών μοντέλων να διορθώνουν προβλήματα και bugs, σε έναν κώδικα με βάση τις περιγραφές των προβλημάτων που τους δίνονται σε φυσική γλώσσα. Δοθείσης μιας περιγραφής αποκλειστικά και μόνο σε φυσική γλώσσα τα μοντέλα δυσκολεύονται να επιλύσουν τα περισσότερα προβλήματα. Το ενδιαφέρον είναι πως συνδυαστικά χρησιμοποιώντας τρία διαφορετικά μοντέλα οι ερευνητές κατάφεραν εν τέλει να λύσουν σχεδόν τα 2/3 των προβλημάτων. Για άλλη μία φορά και στην περίπτωση της διόρθωσης εσφαλμένου κώδικα η χρήση παραδειγμάτων και η χρήση της τεχνικής chain of thought δηλαδή της διαίρεσης του αρχικού προβλήματος σε πιο εύκολα επιλύσιμα προβλήματα αυξάνουν αισθητά την αποτελεσματικότητα των μοντέλων [36], [43].

2 Βιβλιογραφική Επισκόπηση – Θεωρητικό Υπόβαθρο

Όπως και στα περισσότερα είδη προβλημάτων έτσι και στο bug fixing υπάρχει η δυνατότητα χρήσης παραδειγμάτων για την υποβοήθηση του μοντέλου. Μία σκέψη είναι να παρέχουμε αρχικά ένα γενικής φύσεως παράδειγμα bug fixing ακολουθούμενο από ένα πιο συγκεκριμένο παράδειγμα το οποίο σχετίζεται με τα πραγματικά προβλήματα τα οποία θέλουμε να λύσουμε [44]. Για παράδειγμα αντί για την είσοδο «*Ο παρακάτω κώδικας έχει κάποιο λάθος, βρες το και διόρθωσέ το.*» είναι προτιμότερο να γράψουμε «*Ο παρακάτω κώδικας έχει κάποιο λάθος, βρες το και διόρθωσέ το σύμφωνα με το εξής παράδειγμα: 'fruits = ['apple' 'banana' 'cherry'] > fruits = ['apple', 'banana', 'cherry']*».

Κατά την επιχείρηση συγγραφής prompts τα οποία αποσκοπούν στο να δώσουν οδηγίες σε ένα μεγάλο γλωσσικό μοντέλο προκειμένου να εντοπίσει τα λάθη σε αρχεία κώδικα φαίνεται πως καθοριστικό ρόλο κατέχει η συμπερίληψη στις εισόδους των Error log messages τα οποία παρήχθησαν από τα σφάλματα του κώδικα. Τυχόν παράλειψη της παράθεσης αυτών των messages εγκυμονεί τον κίνδυνο της δραματικής μείωσης της αποτελεσματικότητας των μοντέλων στον εντοπισμό των σφαλμάτων. Για παράδειγμα αντί για την είσοδο «*Ο παρακάτω κώδικας έχει κάποιο λάθος, βρες το και διόρθωσέ το.*» είναι προτιμότερο να γράψουμε «*Προσπαθώντας να εκτελέσω τον παρακάτω κώδικα, έλαβα το εξής μήνυμα σφάλματος 'array index out of bounds in line 34', μπορείς να το διορθώσεις;*»

Πέρα από τον απλό εντοπισμό των σφαλμάτων, τα Large Language Models δύνανται να παρέχουν στους προγραμματιστές οδηγίες και κατευθύνσεις οι οποίες αποσκοπούν στο να τους συνδράμουν ώστε να βελτιώσουν την ποιότητα του κώδικα τους, να πραγματοποιήσουν τροποποιήσεις οι οποίες θα βελτιώσουν τον κώδικα τον οποίο έχουν ήδη γράψει ή ακόμα και να διορθώσουν τα σφάλματα τα οποία εμφανίζονται.

Τα μεγάλα γλωσσικά μοντέλα διαθέτουν επίσης τη δυνατότητα συγγραφής επεξηγηματικών περιλήψεων των προγραμμάτων, αλλά και της λειτουργικότητας ξεχωριστών τμημάτων του προγράμματος. Φαίνεται μάλιστα πως με την αξιοποίηση κατάλληλων μεθόδων καταφέρνουν να επιδιορθώσουν την πλειονότητα των σφαλμάτων

και μάλιστα, σε κάποιες περιπτώσεις, ίσως αποτελεσματικότερα ακόμη και από ανθρώπους προγραμματιστές.

Φυσικά πέρα από την παραγωγή κώδικα είναι απαραίτητος και ο έλεγχος του κώδικα αυτού για τυχόν σφάλματα ή κενά ασφαλείας. Είναι εφικτό να ζητήσουμε από ένα μοντέλο τεχνητής νοημοσύνης να δημιουργήσει τεστ τα οποία θα ελέγχουν τον κώδικα και θα εντοπίζουν τα αδύναμα σημεία του. Μία ακόμα ιδιαίτερα πρακτική χρησιμότητα των LLMs είναι η δυνατότητα τους να παράγουν μία ποικιλία από test cases με σκοπό τον έλεγχο και τη διασφάλιση της ευρωστίας και της εγκυρότητας των προγραμμάτων καταφέροντας μάλιστα να εντοπίσουν πληθώρα σφαλμάτων τα οποία είχαν περάσει απαρατήρητα από τους ανθρώπους οι οποίοι επιχείρησαν να ελέγξουν τα ίδια προγράμματα [45].

2.1.11 Η σημασία ενός περιβάλλοντος φιλικού προς τον χρήστη

Πέρα από τον χρήστη ο οποίος φέρει τη μεγαλύτερη ευθύνη για τη διατύπωση ενός πετυχημένου prompt, οι σχεδιαστές της διεπαφής επικοινωνίας με το μοντέλο μπορούν να συνδράμουν τον χρήστη ώστε να τον διευκολύνουν στο έργο του. Για παράδειγμα το σύστημα στο οποίο ο χρήστης διατυπώνει την αρχική είσοδο ως κείμενο μπορεί να λαμβάνει αυτή την είσοδο, να την επεξεργάζεται, να εντοπίζει τα σημαντικότερα στοιχεία της και να αναδιαμορφώνει το κείμενο σε μία πιο refined μορφή η οποία θα προτείνεται ως τελικό prompt εισόδου. Μία άλλη έξυπνη λειτουργία θα ήταν η δυνατότητα συνδυασμού προηγούμενων prompts από το σύστημα σε ένα νέο συνολικό ερώτημα με την αντίστοιχη απάντησή του. Για παράδειγμα ένα σύστημα υποβοήθησης συγγραφής δημιουργικών έργων όπως μυθιστορήματα, θα μπορούσε να συνδυάζει διαφορετικές ιδέες για ένα έργο οι οποίες έχουν διατυπωθεί σε ξεχωριστές εισόδους και να παράγει μία συνολική έξοδο η οποία να προκύπτει από τον συνδυασμό όλων των προηγούμενων prompts εισόδου. Μία άλλη ιδιαίτερα χρήσιμη ιδέα είναι η δυνατότητα αποθήκευσης των prompts και των αντίστοιχων εξόδων ως επαναχρησιμοποιήσιμα εργαλεία σε ένα toolbar. Για παράδειγμα ο χρήστης μπορεί να έχει ήδη δημιουργήσει ένα εργαλείο το οποίο ουσιαστικά είναι ένα prompt που εκτελεί μία συγκεκριμένη δουλειά,

2 Βιβλιογραφική Επισκόπηση – Θεωρητικό Υπόβαθρο

μπορεί για παράδειγμα να παίρνει στην είσοδό του ένα κείμενο και να το να διατυπώνει με πιο λόγιο τρόπο, αυτό το εργαλείο μπορεί να αποθηκευτεί στην πλατφόρμα και να επαναχρησιμοποιηθεί ανά πάσα στιγμή με νέες εισόδους [46].

2.2 Εργαλεία αρωγής στη μηχανική εντολών

2.2.1 Γενική προσέγγιση

Πέρα από την έρευνα και την ανάπτυξη στις τεχνικές συγγραφής κώδικα οι οποίες μπορούν να λύσουν πολλά από τα δομικά προβλήματα του software development, το prompt engineering είναι μία δομημένη αλλά ταυτόχρονα και ευέλικτη μέθοδος για την επίλυση πολυσύνθετων προβλημάτων και τη λήψη εύστοχων απαντήσεων σε πληθώρα διαφορετικών προβλημάτων χωρίς την τροποποίηση του κώδικα και της δομής του software. Για να συνδράμουν τους χρήστες να γράψουν πιο επιτυχημένα prompts έχουν κατασκευαστεί prompt generators τα οποία μπορούν να δημιουργούν εύστοχα και αποτελεσματικά prompts βάσει των οδηγιών των οποίων τους έχει δώσει ο χρήστης. Ουσιαστικά ο χρήστης μπορεί να περιγράψει το πρόβλημα το οποίο αποσκοπεί να λύσει, την προσέγγιση με την οποία θέλει να αντιμετωπιστεί το πρόβλημα από το μοντέλο, την μορφή που θα πρέπει να έχει η απάντηση η οποία εν τέλει θα παραχθεί κ.α. Η συνδρομή τέτοιου είδους δημιουργικών εργαλείων μπορεί να φανεί ιδιαίτερα χρήσιμη σε αρχάριους χρήστες μεγάλων γλωσσικών μοντέλων οι οποίοι ακόμα δεν έχουν πλήρη αντίληψη των δυνατοτήτων του μοντέλου ή των τεχνικών prompting ή επίσης μπορεί να διευκολύνει πιο απαιτητικούς χρήστες οι οποίοι χρησιμοποιούν τα μεγάλα γλωσσικά μοντέλα είτε για την επίλυση μεγάλων προβλημάτων είτε για την ολοκλήρωση περίπλοκων διαδικασιών. Σε αυτή την περίπτωση τα δημιουργικά εργαλεία παραγωγής prompts διευκολύνουν σημαντικά την επεξήγηση των εργασιών και των απαιτήσεών τους [47].

Τα εργαλεία τα οποία συνδράμουν στη δημιουργία prompts σαφώς εφαρμόζονται και για μοντέλα παραγωγής κώδικα. Τα εργαλεία αυτά χωρίς να χρειάζονται πρόσβαση στις παραμέτρους του μοντέλου και χωρίς να προκαλούν αλλαγές στη δομή του, αξιοποιούν απλώς τις πληροφορίες τις οποίες τους παρέχει ο χρήστης. Τέτοιες πληροφορίες μπορεί να είναι κομμάτια κώδικα τα οποία ήδη υπάρχουν, πληροφορίες για τον κώδικα αυτόν, οδηγίες για την εργασία την οποία θέλουμε να επιτελέσει το μοντέλο

καθώς και περιγραφή της μορφής που θα πρέπει να έχει ο κώδικας ο οποίος εν τέλει θα παραχθεί. Ένα παράδειγμα θα ήταν να τροφοδοτήσουμε το μοντέλο με κάποια αρχεία κώδικα ενός προγράμματος στα οποία ίσως υπάρχει ένα κενό, μία έλλειψη δηλαδή κάποιων γραμμών κώδικα, έπειτα θα μπορούσαμε να παρέχουμε στο εργαλείο κάποιες επιπλέον πληροφορίες όπως επεξήγηση της γλώσσας προγραμματισμού, των κλάσεων που υπάρχουν, των συναρτήσεων ή τυχόν συνδέσμων που μπορεί να χρησιμοποιούνται κ.α. Ακόμα θα μπορούσαμε να προσφέρουμε κάποιες οδηγίες όπως πληροφορίες για το σημείο και την περιοχή από όπου λείπουν οι γραμμές κώδικα, πληροφορίες για την λειτουργία του προγράμματος καθώς και τι θα πρέπει να κάνουν οι γραμμές οι οποίες θέλουμε εν τέλει να συμπληρωθούν. Έπειτα, το εργαλείο prompt generation, αξιοποιώντας όλες τις πληροφορίες και τις εισόδους που του δώσαμε θα σχηματίσει ένα προτεινόμενο prompt το οποίο εάν τροφοδοτηθεί στο μεγάλο γλωσσικό μοντέλο κώδικα πρέπει να οδηγήσει στην παραγωγή των επιθυμητών γραμμών κώδικα.

Τέτοιου είδους εργαλεία prompt generation τα οποία ήδη έχουν κατασκευαστεί και έχουν δοκιμαστεί, φαίνεται να βοηθούν σημαντικά στην επιτυχία της διαδικασίας του prompt engineering οδηγώντας εν τέλει στη δημιουργία σημαντικά πιο επιτυχημένων αποτελεσμάτων. Φαίνεται μάλιστα πως αν το εργαλείο prompt generation είναι εξειδικευμένο να συνδράμει μεγάλα γλωσσικά μοντέλα κώδικα μπορεί να οδηγήσει σε πολύ μεγάλη βελτίωση της απόδοσης. Ένα τέτοιο εργαλείο παρουσιάζεται στο [48], το συγκεκριμένο εργαλείο χρησιμοποιεί ένα αποθετήριο και κάποια prompt proposals σε συνδυασμό με τον υπάρχοντα κώδικα καθώς και το σημείο το οποίο θέλουμε να συμπληρωθεί με τον κώδικα τον οποίο εν τέλει θα δημιουργήσει το μεγάλο γλωσσικό μοντέλο ώστε σε συνεργασία με αυτό, στην προκειμένη περίπτωση το Codex, να καταλήξει στην πρόταση ενός εύστοχου prompt το οποίο όταν δοθεί στο Codex θα οδηγήσει στην παραγωγή των επιθυμητών γραμμών κώδικα. Όπως είναι λογικό στην παραγωγή του prompt τον κυρίαρχο ρόλο διαδραματίζει το ίδιο το αρχείο με τον κώδικα στον οποίο θέλουμε να γίνει επέμβαση με δευτερεύουσα βαρύτητα να έχουν συγγενή αρχεία. Ωστόσο φαίνεται πως στην προκειμένη περίπτωση η αξιοποίηση ενός αποθετηρίου κώδικα από το εργαλείο prompt generation είναι το καθοριστικό στοιχείο το οποίο βελτιώνει την απόδοση [48].

2 Βιβλιογραφική Επισκόπηση – Θεωρητικό Υπόβαθρο

Για τις εργασίες συγγραφής κώδικα έχει προταθεί ένα framework το οποίο ονομάζεται Brainstorm και αφορά την αξιοποίηση διαφορετικών σκέψεων της μορφής brainstorming για την υποβοήθηση του μοντέλου στην επίλυση του προβλήματος. Οι σκέψεις αυτές υφίστανται κάποια επεξεργασία και έπειτα παρέχονται στο μοντέλο. Η διαδικασία την οποία ακολουθεί αυτό το framework είναι η εξής, αρχικά έχουμε την φάση του brainstorming όπου πολλά διαφορετικά prompts δημιουργούνται, το καθένα από τα οποία βασίζεται σε μία διαφορετική σκέψη η οποία αποσκοπεί στο να λύσει το αρχικό πρόβλημα. Έπειτα ένα νευρωνικό δίκτυο αξιοποιείται για να κατατάξει τις σκέψεις αυτές με βάση το πόσο πιθανό είναι να οδηγήσουν πράγματι στην λύση του προβλήματος. Και εν τέλει το prompt το οποίο αντιστοιχεί στην προτιμότερη σκέψη τροφοδοτείται στο μοντέλο μαζί με την εκφώνηση του προβλήματος ως υποβοήθηση με τη μορφή υποσημείωσης η οποία εκφράζει τη λογική των ενδιάμεσων βημάτων.

Οι πειραματισμοί δείχνουν πως το framework αυτό συνεισφέρει στην αλγοριθμική συλλογιστική του μοντέλου, με τα ποσοστά επιτυχίας του chat GPT να αυξάνονται σημαντικά και σταθερά. Από την άλλη η υπερβολική σκέψη στα πιο απλά προβλήματα φαίνεται να επιδρά αρνητικά. Κάτι τέτοιο αν και αρχικά ακούγεται παράδοξο μπορεί να εξηγηθεί καθώς το μοντέλο προσπαθώντας να αξιοποιήσει όλη την πληροφορία την οποία του παρέχουμε, αναγκαστικά συγχέει τα φλύαρα τμήματα της εισόδου με αυτά τα οποία φέρουν την περισσότερη βαρύτητα [49].

Στο [50] παρουσιάζεται το εργαλείο Autoprompt το οποίο υπόσχεται τη δημιουργία εξειδικευμένων prompts για συγκεκριμένες εργασίες σε Masked Language Models. Το εργαλείο λαμβάνει ένα αρχικό prompt το οποίο υφίσταται επεξεργασία βάσει ενός template και τροφοδοτείται στο μοντέλο. Η έρευνα ισχυρίζεται πως τα prompts τα οποία κατασκευάζονται από το Autoprompt είναι πιο αποτελεσματικά από τις κοινές εισόδους σε εργασίες όπως το sentiment analysis ακόμα και με λιγοστά ή ελλιπή δεδομένα εισόδου, σε εργασίες natural language inference όπου μάλιστα φαίνεται να υπερέχει σημαντικά από τις εναλλακτικές, σε εργασίες τύπου fact retrieval το Autoprompt φαίνεται να υπερέχει συγκριτικά με τις πιο απλοϊκές μεθόδους όπως η χειροκίνητη συγγραφή της εισόδου. Τέλος, στα πειράματα σε προβλήματα τύπου relation extraction, τα Masked Language Models σε συνεργασία με το Autoprompt καταφέρνουν

να ξεπεράσουν τους ανταγωνιστές τους οι οποίοι ήταν μοντέλα μάθησης με επίβλεψη [50].

Ο άνθρωπος μπορεί να βοηθηθεί στη διαδικασία της συγγραφής prompts από εργαλεία όπως τα prompt generation και prompt engineering tools. Αυτά τα εργαλεία έχουν αναδειχθεί ως απαραίτητοι αρωγοί για πολλούς χρήστες, απαιτητικούς, αρχάριους αλλά και οποιονδήποτε χρήστη ο οποίος επιθυμεί να αποκτήσει αξιόπιστη πρόσβαση σε πετυχημένα prompts εύκολα και γρήγορα. Αυτά τα εργαλεία φέρουν ανυπολόγιστη πρακτική βαρύτητα καθώς παρέχουν στους χρήστες δημιουργικές και δομημένες προσεγγίσεις στην συγγραφή αποτελεσματικών prompts. Για να το πετύχουν αυτό τα διάφορα εργαλεία αξιοποιούν διαφορετικές νοοτροπίες, μεθοδολογίες, και τεχνικές προσεγγίσεις οι οποίες μπορεί να περιλαμβάνουν επεξεργασία φυσικής γλώσσας, μηχανική μάθηση, αλληλεπίδραση με το γραφικό περιβάλλον κ.α. προκειμένου να αντιληφθούν τις επιθυμίες του χρήστη, να τις οργανώσουν και να τις διατυπώσουν σε μορφή πιο δομημένη, ακριβή και εύστοχη. Κατ' αυτόν τον τρόπο οδηγούμαστε σε εισόδους στο μοντέλο οι οποίες δύνανται να προσφέρουν σημαντικά οφέλη όπως ενίσχυση της απόδοσης του μοντέλου, ελαχιστοποίηση των προκαταλήψεων και εξατομικευμένη εμπειρία χρήστη. Επιπλέον μπορούν να σταθούν ιδιαίτερα χρήσιμα στο κοινωνικό σύνολο καθώς προσφέρουν τη δυνατότητα αξιοποίησης της χρήσης των μεγάλων γλωσσικών μοντέλων ακόμα και σε άτομα χωρίς τεχνολογική κατάρτιση και εμπειρία συμβάλλοντας στην οριζόντια αλλά και κάθετη επέκταση της χρήσης αυτών των μοντέλων για περισσότερους σκοπούς και από περισσότερα κοινωνικά στρώματα.

2.2.2 Εργαλεία prompt generation

Όσον αφορά τα εργαλεία prompt generation τα οποία δημιουργούν το prompt από την αρχή, τα περισσότερα χαρακτηρίζονται από ιδιαίτερα απλοϊκή λειτουργικότητα η οποία περιορίζεται στο σταδιακό χτίσιμο του prompt λέξη προς λέξη πατώντας κάποια από τα προκαθορισμένα κουμπιά του γραφικού περιβάλλοντος. Για παράδειγμα εάν θέλουμε να γράψουμε ένα ποίημα για μία γάτα θα επιλέγαμε αντιστοίχως τα κουμπιά "write" έπειτα "poem" και τέλος "cat" να δημιουργηθεί το prompt: "*Write a poem about*

2 Βιβλιογραφική Επισκόπηση – Θεωρητικό Υπόβαθρο

a cat". Είναι εμφανές ωστόσο πως η συγκεκριμένη προσέγγιση πέραν του ότι είναι ιδιαίτερα περιοριστική δεν προσφέρει και κανένα ουσιαστικό όφελος στον χρήστη.

Ένα πιο ενδιαφέρον εργαλείο prompt generation είναι το Neural Writer. Αυτό το εργαλείο δέχεται ως είσοδο την περιγραφή της ζητούμενης απάντησης από το μεγάλο γλωσσικό μοντέλο και αξιοποιώντας επεξεργασία φυσικής γλώσσας σε συνδυασμό με μία βιβλιοθήκη από prompts κατασκευάζει από την αρχή μία ολοκληρωμένη, εύστοχη και περιγραφική είσοδο για το chatGPT. Για παράδειγμα εάν το ζητούμενο αποτέλεσμα το οποίο ορίσουμε είναι: "*a poem about a cat falling in love with the dog*", το prompt το οποίο θα παραχθεί ως συνιστώμενη είσοδος στο chatGPT θα είναι το εξής: "*Generate a poem where a cat develops feelings for a dog and falls in love*" [51].

Άλλες προσεγγίσεις για εργαλεία prompt generation τα οποία ωστόσο δεν είναι ακόμα διαθέσιμα στο κοινό, είναι το customization του prompt με βάση τις απαιτήσεις που θα ορίσει ο χρήστης μέσω γραπτού λόγου ή με βάση τα keywords τα οποία θα επιλέξει από μία προκαθορισμένη λίστα χαρακτηριστικών όπως το ύφος του λόγου, ο χαρακτήρας τον οποίο θα πρέπει να υιοθετήσει το μοντέλο, η μορφή της απάντησης κ.α. Ακόμη μία άλλη ιδέα είναι η δημιουργία πολλών παραλλαγών και εναλλακτικών εισόδων για την ίδια εργασία, ώστε ο χρήστης να μπορεί να δοκιμάσει διαφορετικά prompts και να επιλέξει αυτό το οποίο τον ικανοποιεί περισσότερο.

2.2.3 Εργαλεία prompt engineering

Τα εργαλεία prompt engineering διαφέρουν στο ότι αντί να δημιουργούν μόνα τους το prompt από την αρχή δέχονται ως είσοδο ένα υπάρχων prompt το οποίο τροποποιούν και βελτιστοποιούν σύμφωνα με τις κατευθύνσεις τις οποίες τους παρέχει ο χρήστης. Τα περισσότερα εργαλεία αυτού του τύπου παρέχουν φτωχή λειτουργικότητα καθώς περιορίζονται απλώς στο να συγκρίνουν διαφορετικά prompts σε διαφορετικά μοντέλα και ίσως με διαφορετικές παραμέτρους όπως η τυχαιότητα της απάντησης.

Ένα εισαγωγικό παράδειγμα ενός εργαλείου prompt engineering είναι το Eye for AI. Αυτό το εργαλείο δέχεται ως είσοδο ένα αρχικό prompt το οποίο γράφει ο χρήστης και το οποίο μπορεί να συνοδευτεί από κάποιες επιπλέον πληροφορίες τις οποίες μπορεί

ο χρήστης να παρέχει στο εργαλείο επιλέγοντας συγκεκριμένα χαρακτηριστικά της ζητούμενης απάντησης από μία προκαθορισμένη λίστα. Το συγκεκριμένο εργαλείο αφορά την παραγωγή εικόνων οπότε η λίστα περιλαμβάνει χαρακτηριστικά όπως το μέγεθος, η τεχνοτροπία, το ύφος, η αισθητική, το εικαστικό ρεύμα κ.α. τα οποία συμπληρώνουν την αρχική είσοδο. Ακόμα ο χρήστης μπορεί να εισάγει μία ενδεικτική εικόνα ως παράδειγμα. Παρόμοια λογική και μεθοδολογία ακολουθεί και το εργαλείο Taskade το οποίο δέχεται ως είσοδο ένα αρχικό prompt γραμμένο από τον χρήστη και έπειτα του ζητάει να το συνοδέψει επιλέγοντας από μία προκαθορισμένη λίστα την προσέγγιση η οποία θα πρέπει να υιοθετηθεί για την επίλυση του συγκεκριμένου προβλήματος όπως brainstorming, question answering κ.α. [52].

Μία πολύ ενδιαφέρουσα προσέγγιση είναι αυτή η οποία υλοποιείται με το εργαλείο FlowGPT. Το εργαλείο αυτό δεν στοχεύει αποκλειστικά στο prompt engineering αλλά στη δημιουργία και δημοσίευση διαλογικών chatbots βασισμένων στο chatGPT. Κάποια από αυτά τα chatbots είναι εξειδικευμένα στο prompt engineering. Αυτή η προσέγγιση παρέχει μεγάλη πρακτική ευκολία για άτομα τα οποία στερούνται τεχνικής κατάρτισης στο αντικείμενο το οποίο αφορά η εργασία. Εάν για παράδειγμα, κάποιος θέλει να γράψει κώδικα χωρίς να είναι προγραμματιστής μπορεί εύκολα να οδηγηθεί στη δημιουργία ενός αποτελεσματικού prompt περιγράφοντας στο chatbot τους στόχους του μέσω συζήτησης με χρήση φυσικής γλώσσας. Τα chatbots αυτά συνήθως είναι εξειδικευμένα στην παραγωγή prompts για ένα συγκεκριμένο είδος εργασίας κάτι το οποίο σημαίνει πως οι προτεινόμενες είσοδοι οι οποίες παράγονται είναι αρκετά λεπτομερείς και περιγραφικές. Ωστόσο η συζήτηση τυπικά είναι μονολιθική και μονοδιάστατη καθώς τα περισσότερα chatbots δεν κάνουν διευκρινιστικές ερωτήσεις, ούτε παρέχουν συστάσεις παρά μόνο απαντάνε ψυχρά στα αιτήματα του χρήστη με ένα ή περισσότερα εναλλακτικά prompts για το πρόβλημα το οποίο τέθηκε. Αυτό σημαίνει πως τα chatbots συχνά αυτοσχεδιάζουν συμπληρώνοντας μόνα τους, αυθαίρετα χαρακτηριστικά στην περιγραφή του επιθυμητού αποτελέσματος ενώ ιδανικά θα έπρεπε να τα ζητούν από τον χρήστη [53].

Το εργαλείο promptGen αξιοποιεί ένα παραγωγικό νευρωνικό δίκτυο προκειμένου να δημιουργήσει δυναμικά prompts βάσει της εισόδου που του παρέχει ο

2 Βιβλιογραφική Επισκόπηση – Θεωρητικό Υπόβαθρο

χρήστης. Με αυτό τον τρόπο βοηθάει στο να ανακαλύψουμε τις γνώσεις ενός μοντέλου πάνω σε ένα συγκεκριμένο θέμα, να θέσουμε εύστοχες ερωτήσεις οι οποίες θα γίνουν εύκολα κατανοητές από το μοντέλο, να ζητήσουμε από το μοντέλο να συντάξει δημιουργικό κείμενο ή ακόμα μπορούμε να αξιοποιήσουμε το promptGen προκειμένου να παράγουμε πολλά διαφορετικά prompts για την ίδια εργασία σε περίπτωση που αυτό είναι απαραίτητο ή σε περίπτωση που επιθυμούμε να τα συγκρίνουμε και να επιλέξουμε το πιο αποτελεσματικό [54].

Υπάρχουν εργαλεία prompt engineering όπως το PromptRefine και το prompts.ai, τα οποία μας επιτρέπουν να εισάγουμε μεταβλητές μέσα στο prompt τις οποίες αργότερα μπορούμε να ορίσουμε σε όποια τιμή επιθυμούμε. Τέτοιου είδους προσεγγίσεις μπορούν να φανούν χρήσιμες σε περιπτώσεις όπου για παράδειγμα επιθυμούμε να γράψουμε κώδικα ο οποίος θα δέχεται από τον χρήστη ως είσοδο μία ή περισσότερες λέξεις και έπειτα θα τις εισάγει ως μεταβλητές στο προκαθορισμένο prompt. Με αυτό τον τρόπο μπορούμε να παράγουμε πολλά διαφορετικά prompts προσαρμοσμένα στις επιθυμίες του κάθε ξεχωριστού χρήστη τα οποία όμως όλα βασίζονται στον ίδιο σκελετό [55].

Ένα πιο ολοκληρωμένο εργαλείο είναι το Prompter. Αυτό το εργαλείο μας επιτρέπει να εισάγουμε στο prompt πέρα από μεταβλητές και συναρτήσεις. Οι συναρτήσεις αυτές αντιστοιχούν στις συναρτήσεις του chatGPT επομένως ορίζονται και χρησιμοποιούνται με τον ίδιο τρόπο. Αρχικά ορίζουμε το όνομα και την περιγραφή της συνάρτησης τα οποία ακολουθούνται από κώδικα JSON όπου παρέχεται η περιγραφή των παραμέτρων. Παράλληλα μπορούμε να πειραματιστούμε με διάφορες άλλες παραμέτρους του μοντέλου, όπως η θερμοκρασία, δηλαδή η τυχαιότητα του αποτελέσματος το οποίο μπορεί να μεταφραστεί και ως το επίπεδο δημιουργικότητας του μοντέλου όπου υψηλότερη θερμοκρασία σημαίνει υψηλότερη δημιουργικότητα. Παρομοίως μπορούμε να ορίσουμε το επίπεδο nucleus sampling, δηλαδή την τυχαιότητα με την οποία επιλέγεται κάθε νέο token της εξόδου. Ακόμα υπάρχει η δυνατότητα να ορίσουμε το πλήθος των παραγομένων αποτελεσμάτων το οποίο σημαίνει ότι μπορούμε να λάβουμε περισσότερες από μία απαντήσεις για το ίδιο ερώτημα προκειμένου να επιλέξουμε την προτιμότερη. Σημειώνουμε ότι σε περιπτώσεις όπου έχουμε επιλέξει την παραγωγή πολλαπλών αποτελεσμάτων συστήνεται η επιλογή θερμοκρασίας πάνω από το μηδέν. Επίσης μπορούμε να αξιοποιήσουμε τη δυνατότητα του frequency και presence

penalty, δηλαδή tokens τα οποία έχουν ήδη εμφανιστεί πολλές φορές στο κείμενο θα έχουν όλο και μικρότερη πιθανότητα να επανεμφανιστούν, κάτι το οποίο αποτελεί άλλον έναν τρόπο να επηρεάσουμε τη δημιουργικότητά του μοντέλου. Τέλος, μπορούμε να ορίσουμε και έναν μέγιστο αριθμό tokens για την έξοδο. Το εργαλείο αυτό αποτελεί μία αξιολογη προσπάθεια πειραματισμού με τις τεχνικές παραμέτρους, όχι μόνο του prompt engineering αλλά και της λειτουργίας του μοντέλου [56].

Παρόμοια λειτουργικότητα με το Prompter παρέχει και το εργαλείο BetterPrompt με την επιπλέον δυνατότητα σύγκρισης των prompts ανάλογα με τις παραμέτρους τους [57].

Ένα επίσης πολύ ενδιαφέρον και αρκετά πλήρες εργαλείο είναι το promptKnit. Το εργαλείο αυτό βοηθάει τους χρήστες στο να καταλήξουν σε πιο εύστοχα αποτελέσματα αξιοποιώντας τη μέθοδο few-shot. Όταν ο χρήστης ξεκινά μία συνομιλία, αφότου εισάγει τις οδηγίες τις οποίες θα πρέπει να ακολουθήσει το σύστημα, το promptKnit παράγει αυτόματα 3 τυχαία παραδείγματα από διαφορετικά prompts μαζί με τις αντίστοιχες απαντήσεις του μοντέλου. Στη συνέχεια ο χρήστης μπορεί να τροποποιήσει τα prompts και τις απαντήσεις τους ώστε να ακολουθούν την λογική την οποία επιθυμεί να επιβάλει στο μοντέλο και έπειτα να τα εισάγει όλα μαζί ως γενικά παραδείγματα στο σύστημα. Αυτό αποτελεί έναν ιδιαίτερα εύκολο, γρήγορο και αποτελεσματικό τρόπο εφαρμογής της μεθόδου few-shot. Παράλληλα το εργαλείο αυτό συμπεριλαμβάνει πολλές από τις δυνατότητες του Prompter όπως ρύθμιση θερμοκρασίας, nucleus sampling, χρήση μεταβλητών και συναρτήσεων και επιλογή μεγέθους της εξόδου σε token. Επιπλέον μία ακόμα δυνατότητα την οποία μας παρέχει το promptKnit είναι η άμεση σύγκριση των prompts και των αντίστοιχων αποκρίσεων βάσει των διαφορετικών ρυθμίσεων ή εντολών υπό τις οποίες διαμορφώθηκαν [58].

Μία ακόμα πολύ χρήσιμη δυνατότητα είναι αυτή η οποία παρέχεται από το Wale η οποία αφορά την εφαρμογή prompts σε σύνολα δεδομένων. Το εργαλείο αυτό επιτρέπει στον χρήστη τη φόρτωση συνόλων δεδομένων στην πλατφόρμα του, έπειτα μπορούμε να εισάγουμε το όνομα κάθε στήλης από το σύνολο δεδομένων ως μία μεταβλητή στην είσοδο του γλωσσικού μοντέλου. Για παράδειγμα εάν έχουμε ένα

2 Βιβλιογραφική Επισκόπηση – Θεωρητικό Υπόβαθρο

σύνολο προτάσεων τις οποίες θέλουμε να μεταφράσουμε από τα Αγγλικά σε διάφορες ξένες γλώσσες, μπορούμε να θεωρήσουμε ως μία στήλη του πίνακα τη γλώσσα στην οποία επιθυμούμε να μεταφράσουμε την κάθε πρόταση και στην επόμενη στήλη τις προτάσεις καθ' αυτές. Έπειτα θα γράψουμε ένα prompt στο οποίο θα ζητάμε από το μοντέλο να πραγματοποιήσει τη μετάφραση και το οποίο θα περιλαμβάνει δύο μεταβλητές, η μία θα αντιστοιχεί στην εκάστοτε γλώσσα και η άλλη θα αντιστοιχεί στην εκάστοτε πρόταση. Ένα τέτοιο prompt θα μπορούσε να είναι το εξής: "Translate the sentence: '{{sentence}}' from English to {{language}}." Κατά την εκτέλεση, το Wale θα εφαρμόσει αυτόματα το prompt διαδοχικά σε κάθε γραμμή του συνόλου δεδομένων ξεχωριστά, περνώντας κάθε φορά τις κατάλληλες τιμές στις μεταβλητές και θα επιστρέψει το σύνολο των απαντήσεων. Με αυτήν την μεθοδολογία μπορούμε να επεξεργαστούμε πολύ αποτελεσματικά, εύκολα και γρήγορα, έναν μεγάλο όγκο δεδομένων αξιοποιώντας τα μεγάλα γλωσσικά μοντέλα χωρίς να χρειάζεται να εκτελέσουμε επανειλημμένα το ίδιο prompt για κάθε γραμμή του πίνακα [59].

Η δυνατότητα εισαγωγής συνόλων δεδομένων, ωστόσο με διαφορετική νοοτροπία παρέχεται και από το εργαλείο Promptmetheus. Το εργαλείο αυτό διαχωρίζει την είσοδο του μοντέλου σε blocks και δίνει στον χρήστη τη δυνατότητα να εισάγει σε κάθε block δεδομένα όπως κείμενο ή δεδομένα από ένα σύνολο το οποίο έχει ανεβάσει στην πλατφόρμα. Για παράδειγμα εάν επιθυμούμε να εκτελέσουμε sentiment analysis για ένα άρθρο, αντί να αντιγράψουμε ολόκληρο το άρθρο στην είσοδο του μοντέλου μπορούμε απλώς να το ανεβάσουμε ως ένα σύνολο δεδομένων στην πλατφόρμα και να το επιλέξουμε για εισαγωγή στο επιθυμητό block. Έπειτα το Promptmetheus θα εισάγει αυτόματα το κείμενο του άρθρου στο σημείο όπου έχουμε ορίσει και όταν ολοκληρώσουμε τη διαδικασία θα συντάξει το τελικό prompt ενοποιώντας όλα τα blocks. Το εργαλείο αυτό αποτελεί μία εύχρηστη λύση σε περιπτώσεις όπου θέλουμε να ανεβάσουμε ένα μεγάλο αρχείο ως είσοδο στο μοντέλο [60].

Πέρα από τα προαναφερθέντα υπάρχουν και εργαλεία τα οποία αποσκοπούν περισσότερο στην οργάνωση και την αξιολόγηση των prompts παρά στη δημιουργία καινούργιων. Ένα τέτοιο εργαλείο είναι το PromptSource στις λειτουργίες του οποίου συμπεριλαμβάνονται η δυνατότητα δοκιμής prompts τα οποία είτε δημιουργεί ο εκάστοτε χρήστης εκείνη τη στιγμή είτε έχουν δημιουργηθεί από άλλους χρήστες και

έχουν αναρτηθεί στην πλατφόρμα καθώς και η αξιολόγηση τους με διάφορες μετρικές. Το PromptSource παρέχει μία εξειδικευμένη τυποποιημένη γλώσσα αναπαράστασης των prompts στα πλαίσια κάποιων συνόλων δεδομένων. Όσα από αυτά πληρούν τα guidelines της πλατφόρμας μπορούν να αναρτηθούν δημοσίως και να είναι προσβάσιμα από τους υπόλοιπους χρήστες [61].

Ιδιαίτερο ενδιαφέρον παρουσιάζει και μία ξεχωριστή κατηγορία εργαλείων τα οποία βοηθούν τον χρήστη να χτίσει μία αλυσίδα από βήματα επικοινωνίας με μεγάλα γλωσσικά μοντέλα. Σε μία τέτοια αλυσίδα, σε κάθε βήμα το προηγούμενο αποτέλεσμα τίθεται είτε αυτούσιο είτε έπειτα από επεξεργασία ως νέα είσοδος σε ένα μοντέλο ώστε μετά από μία αλληλουχία βημάτων να λάβουμε το τελικό αποτέλεσμα. Τέτοια εργαλεία τα οποία μας βοηθάνε να κατασκευάσουμε αυτού του είδους τις αλυσίδες είναι το PromptSapper και το PromptChainer. Το πρώτο παρέχει μία γραφική διεπαφή στον χρήστη από την οποία μπορούμε εύκολα μέσω της λογικής Drag and Drop να χτίσουμε μία αλληλουχία από API calls σε μεγάλα γλωσσικά μοντέλα. Το κάθε call μπορεί να φέρει τα δικά του δεδομένα τα οποία πιθανόν να παρήχθησαν από κάποιο προηγούμενο. Έτσι μπορούμε να σχηματίσουμε μία ολοκληρωμένη υπηρεσία βασισμένη σε μία υποδομή software χρήσει τεχνητής νοημοσύνης. Παράλληλα υπάρχει η δυνατότητα επεξεργασίας του πραγματικού κώδικα αν και οι δημιουργοί της εφαρμογής εστιάζουν περισσότερο στην ευκολία χρήσης του drag and drop interface [62].

Παρόμοιες δυνατότητες παρέχει και το PromptChainer με τη διαφορά ωστόσο πως σε αντίθεση με το PromptSapper το οποίο προβάλλει ως προμετωπίδα του την αποφυγή συγγραφής κώδικα, στο εργαλείο αυτό κατέχει σημαντικό ρόλο η δυνατότητα του χρήστη να γράψει κώδικα προκειμένου να επεξεργαστεί τα δεδομένα και να αναπτύξει πιο εξειδικευμένες λειτουργίες [63].

Μία πολύ διαφορετική αλλά και πολύ ενδιαφέρουσα προσέγγιση είναι αυτή του εργαλείου DeeperCoder. Το εργαλείο αυτό είναι ένα νευρωνικό δίκτυο το οποίο αποσκοπεί στο να γράψει προγράμματα υπολογιστή. Προκειμένου να το κάνει αυτό δέχεται ως είσοδο τις επιθυμητές εισόδους και εξόδους του προγράμματος που θέτει ο χρήστης, τις περνά μέσα από το δίκτυο και εντέλει δημιουργεί ένα σχήμα λογικής

2 Βιβλιογραφική Επισκόπηση – Θεωρητικό Υπόβαθρο

αφαίρεσης σε μία εξειδικευμένη, domain specific γλώσσα, το οποίο αναπαριστά τις συναρτήσεις οι οποίες θα απαρτίζουν το τελικό πρόγραμμα. Έπειτα ανάλογα με τις πιθανότητες που έχει κάθε συνάρτηση να βρίσκεται στο τελικό πρόγραμμα εκτελεί μία αναζήτηση στον χώρο των πιθανών προγραμμάτων μέχρις ότου να καταλήξει σε αυτό το οποίο ικανοποιεί τα αιτήματα του χρήστη. Το εργαλείο αυτό αν και δε μπορεί να θεωρηθεί ακριβώς prompt engineering, ούτε prompt generation αποτελεί μια καινοτόμα και πρωτοποριακή προσέγγιση στη διαδικασία συγγραφής κώδικα χρήσει νευρωνικών δικτύων [64].

Το Outlines είναι μία βιβλιοθήκη για τη συγγραφή προγραμμάτων τα οποία επικοινωνούν με generative transformers. Το Outlines λειτουργεί ως μία διεπαφή ανάμεσα στον κώδικα και στα παραγόμενα αποτελέσματα του μοντέλου διαχωρίζοντας το prompting από τη λογική δομή του προγράμματος. Με αυτό τον τρόπο βοηθάει τους developers να χτίσουν σταθερές εφαρμογές οι οποίες επικοινωνούν με εξωτερικά συστήματα τεχνητής νοημοσύνης επιτρέποντας την ευκολότερη διαχείριση του μοντέλου ενώ παράλληλα μπορεί να εγγυηθεί ότι τα παραγόμενα αποτελέσματα θα ακολουθούν τις προδιαγραφές και τα πρότυπα που έχει ορίσει ο χρήστης όπως για παράδειγμα το μορφότυπο JSON. Η χρησιμότητα του Outlines έγκειται στο ότι απλοποιεί την επικοινωνία μεταξύ μοντέλου τεχνητής νοημοσύνης και προγράμματος [65].

Το parea.ai είναι μία πλατφόρμα η οποία δίνει στους χρήστες τη δυνατότητα να οργανώνουν, να διαχειρίζονται, να παρακολουθούν και να βελτιώνουν τις εφαρμογές τους οι οποίες βασίζονται σε μεγάλα γλωσσικά μοντέλα μέσω ενός API. Μέσω της πλατφόρμας οι developers μπορούν να πειραματιστούν με διαφορετικά prompts με σκοπό να τα βελτιστοποιήσουν για τις ανάγκες της εφαρμογής τους, να τα δοκιμάσουν μαζικά σε πολλά διαφορετικά test cases προκειμένου να βεβαιωθούν ότι λειτουργούν σωστά σε όλες τις περιπτώσεις και εν τέλει τα ενσωματώσουν μέσω της πλατφόρμας στην εφαρμογή τους. Για την εύκολη επικοινωνία του προγράμματος με την πλατφόρμα παρέχεται στους χρήστες μία βιβλιοθήκη για τη συγγραφή προγραμμάτων και μία διεπαφή μέσω της οποίας οι εφαρμογές τους μπορούν να επικοινωνούν με την πλατφόρμα. Επίσης παρέχεται ένας code editor μέσω του οποίου οι developers μπορούν να αναλύσουν σε βήματα τις αλυσίδες τεχνητής νοημοσύνης τις οποίες έχουν δημιουργήσει προκειμένου πραγματοποιήσουν ελέγχους αποσφαλμάτωσης στις

εφαρμογές τους χωρίς να απαιτείται η επανειλημμένη επανεκτέλεσή τους αφού τα αποτελέσματα κάθε βήματος αποθηκεύονται ως cache [66].

Το dust.tt αποτελεί μία πλατφόρμα μέσα από την οποία ο χρήστης μπορεί να κατασκευάσει τους δικούς του assistants ανάλογα με τη δουλειά την οποία επιθυμεί να εκτελέσουν και με τα δεδομένα τα οποία θα πρέπει να διαχειριστούν. Στην πλατφόρμα προκειμένου να δημιουργήσουμε ένα βοηθό μπορούμε απλώς να συμπληρώσουμε τις οδηγίες που θα πρέπει να ακολουθήσει, να επιλέξουμε το μοντέλο στο οποίο θα βασίζεται και ακόμα μπορούμε να επιλέξουμε το πόσο δημιουργικός θέλουμε να είναι. Επίσης μπορούμε να ορίσουμε τα δεδομένα στα οποία θα έχει πρόσβαση ο βοηθός καθώς και τις εργασίες τις οποίες θα επιτελεί με αυτά όπως αναζήτηση ή τροποποίηση. Ως πηγές δεδομένων μπορούμε να έχουμε στατικά αρχεία τα οποία έχουμε ανεβάσει στην πλατφόρμα ή ακόμα και δυναμικές πηγές όπως το Google Drive, το Github κ.α.

Η πιο χρήσιμη δυνατότητα της πλατφόρμας ωστόσο είναι η δημιουργία εφαρμογών. Ο χρήστης μπορεί να δημιουργήσει εφαρμογές οι οποίες βασίζονται στα μοντέλα τεχνητής νοημοσύνης και στα σύνολα δεδομένων της επιλογής του. Μέσω της πλατφόρμας η εφαρμογή μπορεί να επικοινωνήσει με το διαδίκτυο, να λάβει και να επεξεργαστεί μαζικά όγκους δεδομένων και να παράγει εξόδους. Ενώ οι περισσότερες από αυτές τις λειτουργίες μπορούν να οριστούν εύκολα μέσω του γραφικού περιβάλλοντος της πλατφόρμας, υπάρχει και η δυνατότητα προσθήκης κώδικα. Τέλος, για την επικοινωνία με την εφαρμογή εκτός της πλατφόρμας μας παρέχεται και ένα API. Το dust.tt αποτελεί μία από τις πιο εύχρηστες και ολοκληρωμένες επιλογές για τη δημιουργία μιας εφαρμογής η οποία βασίζεται στην τεχνητή νοημοσύνη. Παράγοντες οι οποίοι το ξεχωρίζουν είναι η εύχρηστη γραφική διεπαφή, οι ποικίλες πηγές δεδομένων τις οποίες δέχεται, οι προηγμένες δυνατότητες επεξεργασίας των δεδομένων όπως το map reduce και η μαζική εκτέλεση prompts πάνω σε σύνολα δεδομένων, ενώ οι επιπλέον δυνατότητες όπως η επικοινωνία με το διαδίκτυο και η δημιουργία βρόγχων είναι στοιχεία τα οποία σπανίζουν στον ανταγωνισμό [67].

Εργαλεία όπως τα Promptist και SmartGPT είναι απλοϊκά στην λογική τους όμως με μεγάλη πρακτική σημασία. Το έργο τους έγκειται στο να λαμβάνουν αρχικά prompts

2 Βιβλιογραφική Επισκόπηση – Θεωρητικό Υπόβαθρο

από τους χρήστες και χρήσει μεγάλων γλωσσικών μοντέλων να τα αναδιαμορφώνουν και να τα αναδιατυπώνουν με τρόπο πιο αναλυτικό, περιγραφικό και εύστοχο. Για παράδειγμα θα μπορούσαν να δεχτούν την αρχική είσοδο: "*A rabbit wearing a space suit*" και να την αναδιαμορφώσουν στην πιο εμπειριστατωμένη μορφή: "*Rabbit is wearing a space suit, digital art, Greg rutkowski, trending cinematographic artstation*". Αυτή η λειτουργικότητα δεν είναι καθόλου ασήμαντη, καθώς παρά τις φαινομενικά λίγες δυνατότητες που μας προσφέρουν τα εργαλεία, η αναδιατύπωση μίας εισόδου με πιο εύστοχο τρόπο ανήκει στον 'σκληρό πυρήνα' της τέχνης του prompt engineering [68], [69].

Το PromptPerfect είναι ένα πολυδιάστατο εργαλείο για την συγγραφή, επεξεργασία και βελτίωση των prompts, την αποθήκευση τους σε μία βάση δεδομένων, τη δημοσίευση τους ως υπηρεσία, τη δημιουργία συναρτήσεων, τη σύγκριση μεταξύ διαφορετικών μοντέλων, τη δημιουργία agents και την προσομοίωση εκτέλεσης των agents σε διάφορες εργασίες και περιβάλλοντα. Αρχικά η πλατφόρμα μας επιτρέπει να εισάγουμε ένα prompt το οποίο αυτόματα αναδιαμορφώνει και αναδιατυπώνει με στόχο να το βελτιώσει ώστε να γίνει πιο αναλυτικό και περιγραφικό. Έπειτα μπορούμε να αναλύσουμε αυτό το prompt βήμα-βήμα. Οι προσεγγίσεις στην επεξεργασία του prompt περιλαμβάνουν την πρόταση βελτιώσεων, τον έλεγχο του αποτελέσματος, τη διαχείριση του μεγέθους του, τη συμμόρφωση με τις καλές πρακτικές και άλλα. Επιπλέον, η πλατφόρμα μας παρέχει ένα οργανωμένο περιβάλλον για την εφαρμογή της τεχνικής few-shot ενώ μάλιστα διαθέτει τη δυνατότητα εισαγωγής ενός αρχείου CSV για την παροχή των παραδειγμάτων. Τα prompts τα οποία μας ικανοποιούν μπορούμε να τα αποθηκεύσουμε σε μία βάση δεδομένων μαζί με τα αποτελέσματά τους.

Μία ακόμα δυνατότητα η οποία μας παρέχεται από το PromptPerfect είναι η αποθήκευση και δημοσίευση των prompts ως υπηρεσίες. Προκειμένου να υλοποιηθεί η υποδομή της υπηρεσίας μπορούμε να δημιουργήσουμε έναν assistant με συγκεκριμένες οδηγίες και μεταβλητές, τις τιμές των οποίων θα ορίζει ο χρήστης κάθε φορά που χρησιμοποιεί την υπηρεσία. Επίσης υπάρχει ένα οργανωμένο περιβάλλον για τον ορισμό και τη διαχείριση συναρτήσεων του μοντέλου GPT, μία 'αρένα' στην οποία μπορούμε να συγκρίνουμε τα αποτελέσματα διαφορετικών μοντέλων για την ίδια είσοδο και τέλος υπάρχει ένα περιβάλλον στο οποίο μπορούμε να δημιουργήσουμε και να δοκιμάσουμε

agents με συγκεκριμένα χαρακτηριστικά, οδηγίες και στόχους, οι οποίοι εκτελούνται σε συγκεκριμένα περιβάλλοντα τα οποία έχουμε δημιουργήσει και διαθέτουν τις δικές τους σαφείς περιγραφές. Για τη δοκιμή των agents, διατίθεται ένα περιβάλλον προσομοίωσης όπου ο χρήστης μπορεί είτε να δημιουργήσει έναν καινούργιο agent με οδηγίες της επιλογής του είτε να επιλέξει έναν ανάμεσα από τους πολλούς που ήδη υπάρχουν. Ακολούθως, μπορεί είτε να δημιουργήσει ένα νέο περιβάλλον με τα χαρακτηριστικά της επιλογής του ή να επιλέξει ένα από τα πολλά που ήδη υπάρχουν. Εν τέλει ο χρήστης επιλέγει δύο ή περισσότερους agents τους οποίους εκτελεί σε ένα συγκεκριμένο περιβάλλον προκειμένου να συγκρίνει τα αποτελέσματά τους. Το PromptPerfect είναι μία πολυσύνθετη πλατφόρμα με πολλές δυνατότητες. Οι ικανότητες που το κάνουν να ξεχωρίζει από τον ανταγωνισμό του όμως είναι η προαγωγή των prompts σε δημόσιες υπηρεσίες και το εργαλείο προσομοίωσης της εκτέλεσης agents [70].

Το PromptIDE είναι ένα εργαλείο για τον έλεγχο της απόδοσης των prompts σε ένα μεγάλο σύνολο διαφορετικών περιπτώσεων κατηγοριοποίησης κειμένου. Το εργαλείο μας παρέχει μεγάλους όγκους από έτοιμα labeled δεδομένα, ο χρήστης αρκεί να γράψει το prompt το οποίο επιθυμεί να δοκιμάσει και να παρέχει τις επιτρεπτές απαντήσεις από το μοντέλο. Κατά την εκτέλεση το εργαλείο θα εφαρμόσει το prompt πάνω στο σύνολο των ετοιμών δεδομένων το οποίο επέλεξε ο χρήστης, θα λάβει τις απαντήσεις για κάθε στοιχείο και θα τις συγκρίνει με τα πραγματικά σωστά labels. Ο χρήστης μπορεί να δοκιμάσει ταυτόχρονα πολλές παραλλαγές του ίδιου prompt εισάγοντας μεταβλητές οι οποίες παίρνουν τιμές από ένα προκαθορισμένο σύνολο. Εν τέλει το εργαλείο αφότου εκτελέσει όλα τα prompts σε όλα τα στοιχεία των δεδομένων θα μας παρουσιάσει τη συνολική απόδοση του κάθε prompt ώστε να επιλέξουμε το πιο αποτελεσματικό. Αφότου επιλέξουμε το προτιμότερο prompt, μπορούμε να το βελτιστοποιήσουμε περαιτέρω. Το εργαλείο μας δίνει τη δυνατότητα να δούμε για ποια labels γίνονται λανθασμένες εκτιμήσεις και να τροποποιήσουμε είτε την είσοδο του μοντέλου είτε το σύνολο των δυνατών απαντήσεων ώστε να αυξηθεί το ποσοστό των επιτυχών εκτιμήσεων. Το PromptIDE είναι ένα από τα πιο εύστοχα και χρήσιμα εργαλεία στη διαμόρφωση αποδοτικών prompts σε εργασίες τύπου classification [71].

3 Μεθοδολογία

Με βάση τα συμπεράσματα από την έρευνα της βιβλιογραφίας στο πρακτικό κομμάτι της εργασίας, επιχειρείται η αξιοποίηση των πορισμάτων του θεωρητικού τμήματος προκειμένου να δημιουργηθούν εργαλεία τα οποία αποσκοπούν στην περαιτέρω αξιοποίηση των μεγάλων γλωσσικών μοντέλων πέρα από τον συνήθη τρόπο χρήσης τους, στην αποδοτικότερη εκμετάλλευση και ενσωμάτωση τους σε ολοκληρωμένα συστήματα και στην προσαύξηση της παραγωγικότητάς τους. Για να το επιτύχουν αυτό, τα εργαλεία, θα λειτουργήσουν ως μεσάζοντες ανάμεσα στο γλωσσικό μοντέλο GPT της OpenAI και στον τελικό χρήστη συνεισφέροντας σημαντικά από πολλές προσεγγίσεις. Αρχικά τα εργαλεία αυτά φροντίζουν ώστε η επικοινωνία χρήστη και μοντέλου να εναρμονίζεται με τα πλαίσια των συμπερασμάτων της θεωρητικής έρευνας. Ακόμη τα εργαλεία θα παρέχουν επιπλέον λειτουργικότητα η οποία ξεπερνάει τους περιορισμούς του συνήθους τρόπου χρήσης αυτών των μοντέλων χτίζοντας επάνω και γύρω από αυτά. Επιπλέον τα εργαλεία ενσωματώνονται αρμονικά μέσα σε ολοκληρωμένα συστήματα και περιβάλλοντα τα οποία διαθέτουν πλούσια λειτουργικότητα και φιλοξενούν και εξυπηρετούν τον χρήστη καθ' όλη τη διάρκεια της εργασίας του.

Τα εργαλεία αυτά, αρχικά λαμβάνουν από τον χρήστη το μήνυμά του το οποίο επιθυμεί να παρέχει στο μοντέλο και αφού του το επεξεργαστούν ανάλογα με τις απαιτήσεις του εκάστοτε περιβάλλοντος και της εκάστοτε εργασίας, το αποστέλλουν στο μοντέλο. Αυτό, πιθανότατα πρόκειται για μία εξειδικευμένη fine tuned έκδοση του μοντέλου, ειδικά εκπαιδευμένη ώστε να εξυπηρετεί τον χρήστη ανάλογα με την εργασία του με έμφαση στον προγραμματισμό. Έπειτα, αφού ληφθεί η απάντηση, αυτή μπορεί είτε να ενσωματωθεί απευθείας σε κάποιο πρόγραμμα ως κώδικας, είτε να αξιοποιηθεί με τη χρήση των επιπλέον λειτουργιών οι οποίες παρέχονται από τα εργαλεία. Αυτές οι επιπλέον λειτουργίες περιλαμβάνουν ανάλυση, έλεγχο, επεξεργασία, δημοσίευση της απάντησης και άλλα. Επιπρόσθετα, παρέχονται στον χρήστη εξειδικευμένοι ψηφιακοί βοηθοί τον ρόλο των οποίων αναλαμβάνει το fine tuned μοντέλο οι οποίοι θα συμπεριφερθούν διαφορετικά ανάλογα με τις οδηγίες οι οποίες τους έχουν δοθεί ούτως ώστε ο χρήστης να επικοινωνεί πάντοτε με τον κατάλληλο βοηθό ο οποίος θα παρέχει πάντοτε την κατάλληλη υποστήριξη προκειμένου να περατώσει την κάθε εργασία.

3.1 Παρουσίαση των εργαλείων

Το πρακτικό κομμάτι της εργασίας περιλαμβάνει την υλοποίηση μιας ιστοσελίδας καθώς και ενός extension τα οποία βοηθούν τον χρήστη να γράψει κώδικα αξιοποιώντας το μεγάλο γλωσσικό μοντέλο GPT της OpenAI. Η ιστοσελίδα περιλαμβάνει ένα ειδικά σχεδιασμένο διαλογικό περιβάλλον χρήστη αλλά και μία fine tuned έκδοση του νευρωνικού δικτύου, ειδικά σχεδιασμένα ώστε να συνδράμουν αρχάριους αλλά και προχωρημένους προγραμματιστές στη συγγραφή συναρτήσεων ή και ολόκληρων προγραμμάτων κώδικα. Το extension, το οποίο επίσης επικοινωνεί με μία άλλη fine tuned έκδοση του νευρωνικού δικτύου, γράφτηκε χρησιμοποιώντας τη γλώσσα προγραμματισμού Typescript, συγγενής γλώσσα με την Javascript, καθότι αυτήν υποστηρίζεται από το περιβάλλον για την κατασκευή extensions. Η υλοποίηση ενσωματώνεται στο προγραμματιστικό περιβάλλον Visual Studio Code και με ιδιαίτερη ευκολία συνδράμει τον χρήστη σε κάθε πτυχή της συγγραφής κώδικα από την προετοιμασία έως το debugging.

Ο ιστότοπος δημιουργήθηκε χρησιμοποιώντας την no code πλατφόρμα Bubble και χωρίζεται σε δύο σελίδες. Η μία αφορά πιο προχωρημένους χρήστες ενώ η άλλη αφορά προγραμματιστές με μικρότερη ή και καθόλου εμπειρία. Η βασική ιστοσελίδα για τους πιο προχωρημένους χρήστες λειτουργεί βάσει του GPT 4 χρησιμοποιώντας assistants και threads ενώ η δεύτερη ιστοσελίδα για τους αρχάριους χρήστες λειτουργεί με βάση fine tuned μοντέλα βασισμένα στο GPT 3.5.

Ακόμα υπάρχει μία σελίδα credentials στην οποία ο χρήστης μπορεί είτε να κάνει Login εάν είναι ήδη εγγεγραμμένος είτε με κάνει Sign up εάν είναι νέος χρήστης, ακολούθως μπορεί να επιλέξει αν θέλει να χρησιμοποιήσει την σελίδα advanced ή την beginners. Αυτομάτως, θα δημιουργηθεί ο πρώτος του assistant στη σελίδα advanced με τις default οδηγίες που σχολιάσαμε προηγουμένως μαζί με το αντίστοιχο thread ενώ στη σελίδα beginners θα δημιουργηθεί αυτομάτως μία νέα συζήτηση με έναν νέο εικονικό assistant με τις οδηγίες που επίσης σχολιάσαμε προηγουμένως.

Και στις 2 ιστοσελίδες το interface περιλαμβάνει ένα πεδίο διαλογικής αμφίδρομης επικοινωνίας με διάφορες εκδοχές από το μεγάλο γλωσσικό μοντέλο GPT της OpenAI σε μορφή μηνυμάτων. Τα μηνύματα του χρήστη εμφανίζονται στη δεξιά πλευρά του πλαισίου, ενώ οι αποκρίσεις του μοντέλου στην αριστερή.

3.2 Η υλοποίηση του ιστοτόπου

3.2.1 Η υλοποίηση της σελίδας “Advanced”

Η σελίδα των προχωρημένων χρηστών ονομάζεται έτσι διότι η επικοινωνία και οι λειτουργίες της, προσανατολίζονται σε πιο τεχνικούς όρους προκειμένου ο χρήστης να διαθέτει όσο το δυνατόν περισσότερο έλεγχο επί της διαδικασίας και επομένως θα πρέπει να διαθέτει έστω κάποια βασική αντίληψη περί προγραμματισμού. Πιο συγκεκριμένα, ο βοηθός ο οποίος αρχικά ενσωματώνεται στην σελίδα αυτή, φέρει οδηγίες σύμφωνα με τις οποίες εξειδικεύεται στην συγγραφή συναρτήσεων, ωστόσο ο κάθε χρήστης μπορεί εύκολα να τροποποιήσει τις οδηγίες αυτές ώστε να τον προσανατολίσει στην αρωγή σε οποιαδήποτε πτυχή της συνολικής εργασίας. Αυτή την στιγμή η ιστοσελίδα βρίσκεται διαθέσιμη στο <https://mai23016.bubbleapps.io/credentials>.

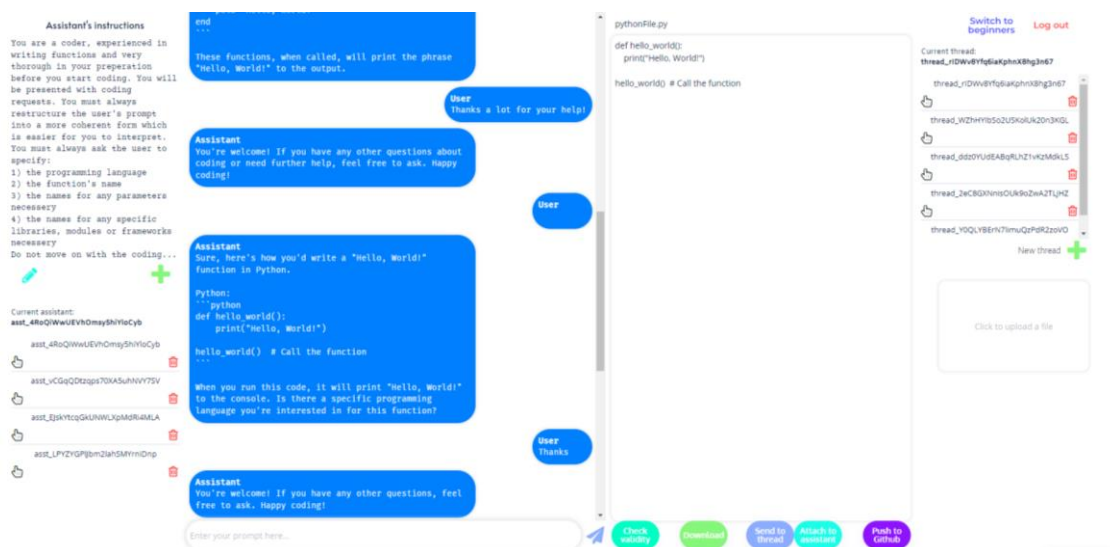
Στην ιστοσελίδα των προχωρημένων χρηστών, αριστερά από το διαλογικό πλαίσιο υπάρχει το πεδίο όπου αναγράφονται οι οδηγίες που έχουν δοθεί στον assistant και από κάτω υπάρχουν δύο πλήκτρα, με το πλήκτρο στα αριστερά εμφανίζουμε ένα pop up στο οποίο μπορούμε να επεξεργαστούμε τις οδηγίες οι οποίες έχουν δοθεί στον συγκεκριμένο assistant, με το πλήκτρο στα δεξιά μπορούμε να δημιουργήσουμε έναν νέο βοηθό ορίζοντας τις οδηγίες με τις οποίες θα ξεκινήσει στο pop up το οποίο θα εμφανιστεί. Ακριβώς από κάτω υπάρχει η λίστα όπου αναγράφονται όλοι οι assistants τους οποίους έχει δημιουργήσει και χρησιμοποιεί ο συγκεκριμένος χρήστης. Πατώντας πάνω στο χεράκι που υπάρχει στην αριστερή μεριά της λίστας μπορούμε να επιλέξουμε τον αντίστοιχο assistant ενώ πατώντας στον κάδο απορριμμάτων που βρίσκεται στην δεξιά μεριά μπορούμε να τον διαγράψουμε. Στο πάνω μέρος της λίστας αναγράφεται ο assistant ο οποίος είναι επιλεγμένος αυτή τη στιγμή.

Ένας assistant ουσιαστικά αποτελεί μία προσωποποίηση του νευρωνικού δικτύου η οποία συμπεριφέρεται με βάση κάποιες συγκεκριμένες οδηγίες που έχουμε ορίσει. Όταν αποστέλλουμε μία είσοδο στο μοντέλο, αυτό θα την επεξεργαστεί σύμφωνα με τον τρόπο τον οποίο έχουμε υποδείξει στις οδηγίες και θα συνθέσει την απάντησή του, πάλι ακολουθώντας τις οδηγίες. Φυσικά ο καθένας μπορεί ανά πάσα στιγμή, να τροποποιήσει τις οδηγίες του κάθε βοηθού τον οποίο χρησιμοποιεί προκειμένου να εξυπηρετεί καλύτερα τις ανάγκες του. Έτσι από την στιγμή της τροποποίησης και μετά ο βοηθός θα συμπεριφέρεται σύμφωνα με τις νέες εντολές, ενώ οι παλαιότερες απαντήσεις του παραμένουν ανεπηρέαστες.

Κάτω από το διαλογικό πλαίσιο μηνυμάτων υπάρχει το πεδίο όπου ο χρήστης μπορεί να γράψει το μήνυμά του προς το μοντέλο και δεξιά από το πεδίο υπάρχει το πλήκτρο με το οποίο μπορεί να το στείλει στο μοντέλο. Στα δεξιά της σελίδας υπάρχει η λίστα με τα threads τα οποία έχει δημιουργήσει ο χρήστης, ουσιαστικά κάθε thread αποτελεί μία ξεχωριστή συζήτηση κατά τη διάρκεια της οποίας το ιστορικό των παλαιότερων μηνυμάτων διατηρείται και χρησιμοποιείται ως context της συζήτησης ενώ ακόμα μπορούμε να αναφερθούμε στα προηγούμενα μηνύματα κατά τη διάρκεια της συνομιλίας. Κατά τη διάρκεια μίας συζήτησης μπορούμε να επικοινωνήσουμε και να λάβουμε μηνύματα από διαφορετικούς assistants μέσα στο ίδιο thread ενώ επίσης μπορούμε να διαθέτουμε πολλά διαφορετικά, παράλληλα threads με τον ίδιο assistant, το καθένα από τα οποία λειτουργεί αυτόνομα, με δικό του context.

Παρομοίως με την αντίστοιχη λίστα των assistants, πατώντας την αριστερή μεριά της λίστας των threads μπορούμε να επιλέξουμε το αντίστοιχο thread ενώ πατώντας τον κάδο απορριμμάτων στα δεξιά μπορούμε να διαγράψουμε το αντίστοιχο thread. Στην κορυφή της λίστας αναγράφεται το thread στο οποίο βρισκόμαστε αυτή τη στιγμή, ακριβώς κάτω από τη λίστα υπάρχει ένα πλήκτρο με το οποίο μπορούμε να δημιουργήσουμε νέα threads.

3 Μεθοδολογία



Εικόνα 1: Το interface της σελίδας "Advanced"

Πιο κάτω υπάρχει ένα πεδίο file uploader από όπου μπορούμε να ανεβάσουμε αρχεία στην ιστοσελίδα, όταν το πατήσουμε θα εμφανιστούν από κάτω τρία πλήκτρα. Το "Cancel" αφορά την ακύρωση του ανεβάσματος του αρχείου σε περίπτωση που το πατήσαμε κατά λάθος, πιέζοντας το πλήκτρο "Attach to assistant" ανεβάζουμε το αρχείο ως δεδομένο του assistant και έπειτα σε κάθε συζήτηση σε κάθε thread όπου επικοινωνούμε με τον συγκεκριμένο assistant μπορούμε να αναφερθούμε στο συγκεκριμένο αρχείο και να του ζητήσουμε να ανατρέξει σε αυτό για μία εργασία. Με αυτήν την επιλογή εάν επιλέξουμε να επικοινωνήσουμε με κάποιον άλλον assistant, ασχέτως του thread στο οποίο βρισκόμαστε, εάν ο βοηθός δεν έχει λάβει το αρχείο ως δεδομένο του δεν θα μπορεί να γνωρίζει για αυτό.

Επίσης, υπάρχει και το πλήκτρο "Send to thread", με το πλήκτρο αυτό ανεβάζουμε το αρχείο στο συγκεκριμένο thread, έτσι μπορεί να χρησιμοποιηθεί από οποιονδήποτε assistant συμμετέχει στη συζήτηση χωρίς ωστόσο αυτό να διατηρείται μόνιμα ως δεδομένο κάποιου assistant ή να επηρεάζει τις υπόλοιπες συζητήσεις. Επομένως στην συγκεκριμένη συζήτηση όλοι οι βοηθοί μπορούν να το αξιοποιήσουν, ωστόσο εκτός αυτής της συζήτησης, κανείς δεν θα γνωρίζει για αυτό. Στην πάνω δεξιά γωνία της σελίδας υπάρχουν τα πλήκτρα "Log out" και "Switch to beginners" τα οποία επιτελούν τις αντίστοιχες λειτουργίες.

Τέλος, ενδιαφέρουσα λειτουργικότητα προσφέρει και το πεδίο που βρίσκεται ανάμεσα στο διαλογικό πλαίσιο και στη λίστα των threads. Το πεδίο αυτό είναι αρκετά μεγάλο έτσι ώστε να μπορούμε να γράψουμε κώδικα είτε να αντιγράψουμε κώδικα τον οποίο παρήγαγε το μοντέλο ούτως ώστε να έχουμε μπροστά μας σε πραγματικό χρόνο το ολοκληρωμένο πρόγραμμα έτσι όπως εν τέλει θα χρησιμοποιηθεί. Σκοπός αυτού του πεδίου, είναι να συνθέσουμε εκεί ολόκληρο το αρχείο κώδικα και να το επεξεργαζόμαστε επιτόπου μέχρι την ολοκλήρωσή του. Πρέπει να θυμόμαστε βέβαια πως εξ αρχής, το περιεχόμενο αυτού του πεδίου δεν αποτελεί δεδομένο κανενός thread, ούτε είναι προσβάσιμο από κανέναν assistant. Ακριβώς από πάνω υπάρχει το πεδίο όπου μπορούμε να συμπληρώσουμε το όνομα του αρχείου του προγράμματος. Κάτω από το πεδίο υπάρχουν κάποια πολύ χρήσιμα πλήκτρα με τα οποία μπορούμε να επιτελέσουμε κάποιες λειτουργίες οι οποίες αφορούν το πεδίο του κώδικα. Αρχικά υπάρχει το πλήκτρο “Download” το οποίο πολύ απλά αποθηκεύει στον υπολογιστή μας το αρχείο κώδικα με το όνομα το οποίο έχουμε ορίσει από πάνω.

Επίσης, υπάρχουν τα πλήκτρα “Sent to thread” και “Send to assistant”, τα οποία μετατρέπουν τον κώδικα σε αρχείο και επιτελούν αντίστοιχες λειτουργίες με τα ομώνυμα πλήκτρα κάτω από τον File uploader. Πιο συγκεκριμένα, το πρώτο πλήκτρο, ανεβάζει το περιεχόμενο του πεδίου με τον αντίστοιχο τίτλο στο context της επιλεγμένης συζήτησης, ενώ το δεύτερο, το επισυνάπτει ως δεδομένο του επιλεγμένου βοηθού. Ακόμα, υπάρχει το πλήκτρο “Push to Github”, το οποίο όταν το πατάμε, εμφανίζεται ένα pop up στο οποίο μας ζητάει να συμπληρώσουμε τέσσερα πεδία, το username μας στο Github το όνομα του repository, το όνομα του αρχείου και το προσωπικό μας authorization key για το API του Github. Τέλος, πατώντας “Upload” το αρχείο ανεβαίνει αυτομάτως στην αντίστοιχη τοποθεσία στο Github με το όνομα το οποίο του δώσαμε, μέσω του API του Github. Σημειώνεται βέβαια πως για να γίνει κάτι τέτοιο, θα πρέπει ο χρήστης να έχει εκδώσει δικό του κλειδί για το API του Github και να έχει ήδη δημιουργήσει το repository στο οποίο επιθυμεί να ανεβάσει το αρχείο. Τέλος υπάρχει και το πλήκτρο “Check validity”, όπου πατώντας το, το πρόγραμμα ανεβαίνει ως αρχείο στο ενεργό thread και γίνεται αυτομάτως αίτημα στον assistant να εξετάσει την συντακτική ορθότητα του κώδικα. Ο βοηθός μόλις λάβει το αρχείο ως δεδομένο του, θα το εξετάσει μέσω της υπηρεσίας Code Interpreter την οποία μας παρέχει η OpenAI. Η υπηρεσία αυτή

3 Μεθοδολογία

δύναται όχι μόνο να ελέγχει τα αρχεία κώδικα για συντακτικά λάθη αλλά μπορεί και να προσομοιώνει την εκτέλεσή τους και να παράγει την τελική έξοδο του προγράμματος. Αφότου λοιπόν ο βοηθός, μέσω αυτής της λειτουργίας ελέγξει τον κώδικα, θα μας ενημερώσει μέσω μηνύματος στο πλαίσιο της συζήτησης εάν εντοπιστήκαν σφάλματα και θα μας συνδράμει στη διόρθωσή τους .

Αρχικά, ο πρώτος assistant ο οποίος δημιουργείται αυτόματα με τη δημιουργία του λογαριασμού ώστε να μπορέσει ο χρήστης να χρησιμοποιήσει την υπηρεσία, φέρει κάποιες προκαθορισμένες, αναλυτικές εντολές οι οποίες αποσκοπούν στο να τον καταστήσουν έναν άρτιο, οργανωτικό, προσεκτικό και επιδέξιο βοηθό συγγραφής κώδικα για συναρτήσεις. Οι εντολές αυτές, αρχικά τον καθοδηγούν να είναι πολύ προσεκτικός πριν αρχίσει να γράφει κώδικα και να αναδιαμορφώνει το αίτημα του χρήστη σε μορφή πιο συνεκτική και εύληπτη από τον ίδιο. Έπειτα θα πρέπει να σιγουρευτεί πως ο χρήστης έχει παράσχει ξεκάθαρες πληροφορίες σχετικά με τη γλώσσα προγραμματισμού, το όνομα της συνάρτησης και των παραμέτρων, τις ακριβείς εισόδους και εξόδους καθώς και την ανάγκη χρήσης συγκεκριμένων βιβλιοθηκών. Ο assistant δεν θα πρέπει να ξεκινήσει να γράφει κώδικα προτού διαβεβαιωθεί ότι όλα τα παραπάνω έχουν καθοριστεί, διαφορετικά θα πρέπει να ζητήσει από τον χρήστη επιπλέον διευκρινίσεις. Έπειτα ο assistant θα πρέπει να εξάγει τη λογική πίσω από το αίτημα του χρήστη και να την καταταμίσει σε μικρά, κατανοητά και εύκολα υλοποιήσιμα βήματα.

Εάν υπάρχουν απορίες σχετικά με την λογική του προγράμματος θα πρέπει να ζητηθούν εξηγήσεις, εάν υπάρχουν ειδικές περιπτώσεις θα πρέπει να ζητηθούν οδηγίες για τη διαχείρισή τους. Κάθε φορά που γίνεται μία ερώτηση ο assistant θα πρέπει να διευκρινίζει τα αίτια που οδήγησαν σε αυτή την απορία. Τέλος, ο κώδικας που θα γραφτεί θα πρέπει πάντα να συμπεριλαμβάνει επεξηγηματικά σχόλια. Όπως αναφέρθηκε παραπάνω οι οδηγίες αυτές αλλά και οι οδηγίες κάθε νέου βοηθού μπορούν να τροποποιηθούν σύμφωνα με τη βούληση του κάθε χρήστη. Ωστόσο, πιο συγκεκριμένα οι αρχικές οδηγίες είναι οι εξής: *«You are a coder, experienced in writing functions and very thorough in your preparation before you start coding. You will be presented with coding requests. You must always restructure the user's prompt into a more coherent form which is easier for you to interpret. You must always ask the user to specify:*

1) the programming language

2) the function's name

3) the names for any parameters necessary

4) the names for any specific libraries, modules or frameworks necessary

Do not move on with the coding before making sure that the user has specified all the above 4.

If you have any doubts regarding the use of any specific library, module or framework, do not move on with the coding before asking the user to elaborate.

You must always extract the reasoning behind the users request and break it into small, easy steps. If you have any doubts regarding the logical structure of the function, do not move on with the coding before asking the user to provide more details.

If there are any special cases, do not move on with the coding before asking the user on how to handle them.

When asking the user for any information, you must always explain what thought of yours led you to asking this question.

The code you write must always include explanatory comments.

You must always provide a thorough analysis of the code you have written.»

3.2.2 Η υλοποίηση της σελίδας “Begginers”

Στην σελίδα beginners υπάρχουν κάποιες διαφορές, κυρίως λόγω του διαφορετικού τρόπου με τον οποίο είναι υλοποιημένη η λειτουργικότητά της. Η λειτουργικότητα είναι υλοποιημένη χρησιμοποιώντας τα Chat completions του API της OpenAI και όχι Assistants αν και γίνεται απόπειρα δημιουργίας ενός εικονικού assistant χρησιμοποιώντας τον ρόλο System των Chat completions. Κάθε chat αυτού του τύπου, αποτελεί ουσιαστικά μία ανεξάρτητη συζήτηση κατά την οποία διατηρείται το ιστορικό των μηνυμάτων που ανταλλάξαμε ως context της συζήτησης και επομένως μπορούμε να αναφερθούμε σε προηγούμενα μηνύματα. Κάθε τέτοιο chat διαθέτει τρεις ρόλους, τον χρήστη, τον εικονικό βοηθό ο οποίος αποκρίνεται στα αιτήματα του χρήστη και τον ρόλο system όπου παρέχονται κάποιες οδηγίες, εντολές και κατευθύνσεις στον εικονικό βοηθό

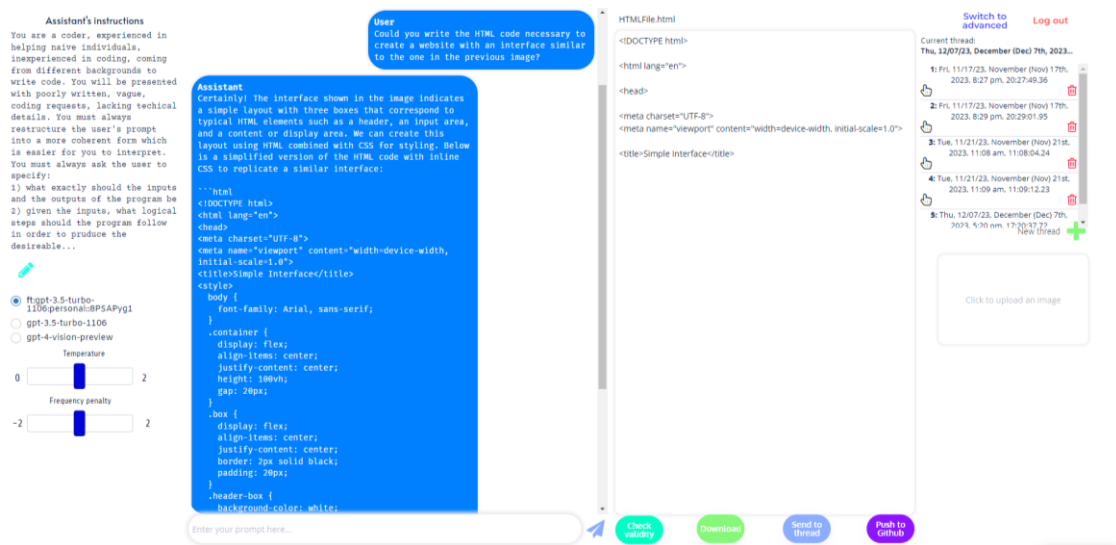
3 Μεθοδολογία

σχετικά με τον τρόπο με τον οποίο θα πρέπει να διαχειρίζεται τα αιτήματα του χρήστη αλλά και με τον τρόπο με τον οποίο θα πρέπει να συνθέτει τις απαντήσεις του.

Πάλι υπάρχει το ίδιο διαλογικό περιβάλλον επικοινωνίας με το μοντέλο και από κάτω το πεδίο εισαγωγής των μηνυμάτων. Στα αριστερά υπάρχει το πεδίο με τις εντολές οι οποίες έχουν δοθεί στον επιλεγμένο εικονικό assistant και από κάτω υπάρχει ένα πλήκτρο για την επεξεργασία των εντολών αυτών. Στην συγκεκριμένη ιστοσελίδα δεν μπορούμε να έχουμε αποθηκευμένους παράλληλα διαφορετικούς assistants.

Κάτω από αυτό το πεδίο βρίσκονται κάποια radio buttons για την επιλογή του επιθυμητού μοντέλου. Οι επιλογές περιλαμβάνουν το GPT 3.5 turbo 1106, δηλαδή το απλό βασικό μοντέλο, το GPT 4 vision preview το οποίο επιτρέπει το ανέβασμα και εικόνων τις οποίες καταφέρει να ‘διαβάσει’ και να αντιληφθεί το περιεχόμενό τους, και το fine tuned GPT 3.5 το οποίο είναι εξειδικευμένο στο να χειρίζεται κλήσεις για τη συγγραφή προγραμμάτων από χρήστες οι οποίοι στερούνται εμπειρίας στον προγραμματισμό.

Ακριβώς από κάτω θα βρούμε sliders αφενός για την επιλογή της θερμοκρασίας ‘temperature’ των απαντήσεων του μοντέλου, δηλαδή του πόσο τυχαίες αλλά ταυτόχρονα και πόσο δημιουργικές θα είναι οι απαντήσεις του μοντέλου και για το frequency penalty δηλαδή το πόσο θέλουμε να περιορίσουμε τις επαναλαμβανόμενες απαντήσεις από το μοντέλο. Εάν προτιμάμε να ενισχύσουμε τη δημιουργικότητα τότε θα πρέπει να ρυθμίσουμε την θερμοκρασία και το frequency penalty σε υψηλότερες τιμές καθότι αυτό σημαίνει πως έτσι θα παράγονται περισσότερα πρωτότυπα tokens, δηλαδή λέξεις και με πιο απρόβλεπτο τρόπο. Απεναντίας, εάν προτιμάμε πιο προβλέψιμες, τυπικές αλλά ταυτόχρονα και αξιόπιστες απαντήσεις, τότε θα πρέπει να ρυθμίσουμε την θερμοκρασία και το frequency penalty σε χαμηλότερες τιμές καθότι αυτό σημαίνει πως έτσι θα επιλέγονται τα tokens τα οποία έχουν τις περισσότερες πιθανότητες να είναι εύστοχα. Στα δεξιά υπάρχει πάλι η λίστα επιλογής των threads, η επιγραφή του ενεργού thread και το πλήκτρο δημιουργίας νέου thread. Από πάνω υπάρχουν και πάλι τα πλήκτρα “Switch to advanced” και “Logout”. Από κάτω, το πεδίο του File uploader έχει αντικατασταθεί από έναν Picture uploader.



Εικόνα 2: Το interface της σελίδας "Beginners"

Λόγω της διαφορετικής υλοποίησης των δύο ιστοσελίδων γίνεται αδύνατη η χρήση εικόνων στην advanced σελίδα αφού δεν επιτρέπεται από την εταιρία η δημιουργία assistant βασισμένου στο GPT 4 vision preview το οποίο είναι το μοναδικό μοντέλο με τη δυνατότητα αντίληψης εικόνων. Αντιστοίχως καθίσταται αδύνατη και η χρήση αρχείων στην σελίδα beginners αφού δεν μπορούμε να ανεβάσουμε αρχεία μέσω της λειτουργίας των Chat completions. Πατώντας τον Picture uploader μπορούμε να επιλέξουμε μία εικόνα από τον υπολογιστή μας και να την παρέχουμε στο μοντέλο κωδικοποιημένη ως Base64. Αφότου επιλέξουμε την εικόνα εμφανίζονται δύο πλήκτρα το πλήκτρο "Cancel" για την ακύρωση του ανεβάσματος και το πλήκτρο "Sent to thread" για την αποστολή της εικόνας στην επιλεγμένη συζήτηση.

Τέλος, ανάμεσα στο διαλογικό πλαίσιο επικοινωνίας με το μοντέλο και στη λίστα των threads έχουμε και πάλι το πεδίο όπου μπορούμε να γράψουμε κώδικα, να κάνουμε copy τον κώδικα τον οποίο παρήγαγε το μοντέλο και να συντάξουμε ένα ολοκληρωμένο πρόγραμμα. Και πάλι στο επάνω μέρος βρίσκουμε το πεδίο εισαγωγής του ονόματος του αρχείου. Για άλλη μία φορά στο κάτω μέρος υπάρχει το πλήκτρο "Push to Github" όπου και πάλι εμφανίζεται το pop up για την εισαγωγή των στοιχείων μας και το ανέβασμα του προγράμματος ως αρχείο στο Github μέσω του API του Github για το οποίο και πάλι θα πρέπει να έχουμε εκδώσει το κατάλληλο κλειδί και να επιλέξουμε ένα υπαρκτό repository. Υπάρχει το πλήκτρο "Download" για την αποθήκευση του προγράμματος ως

3 Μεθοδολογία

αρχείο στον υπολογιστή μας, το πλήκτρο “Send to thread” όπου το πρόγραμμα θα αποσταλεί ως μήνυμα κειμένου στο μοντέλο στα πλαίσια της συγκεκριμένης συζήτησης, ενώ τέλος υπάρχει το πλήκτρο “Check validity” όπου και πάλι το πρόγραμμα θα αποσταλεί ως μήνυμα κειμένου στο μοντέλο και θα του ζητηθεί αυτομάτως να εξετάσει την συντακτική ορθότητα του κώδικα. Αυτή τη φορά ο έλεγχος τους κώδικα ωστόσο δεν θα γίνει μέσω της λειτουργίας του Code interpreter αφού αυτό αφορά αποκλειστικά τους assistants, αντιθέτως ο έλεγχος θα πραγματοποιηθεί απευθείας από το μοντέλο μέσω αυτοματοποιημένου μηνύματος.

Για την εκπόνηση της εκπαίδευσης στα πλαίσια του fine tuning του μοντέλου GPT3,5 turbo 1106, χρησιμοποιήθηκαν κάποια ενδεικτικά, κακώς διατυπωμένα prompts με ασάφειες, ελλείψεις και πλήρη έλλειψη από τεχνικές λεπτομέρειες. Τέτοια prompts θα μπορούσαν να γράψουν άτομα τα οποία στερούνται εμπειρίας στον προγραμματισμό και τεχνική κατάρτιση, ενώ το είδος των αιτημάτων διαφοροποιείται έτσι ώστε να καλύψει το ενδεχόμενο χρήσης του εργαλείου από άτομα προερχόμενα από διαφορετικά περιβάλλοντα. Αφότου δημιουργήθηκαν αρκετά διαφορετικά παραδείγματα με αυτού του είδους τα prompts και αυτής της μορφής τις αποκρίσεις, το μοντέλο τροφοδοτήθηκε με αυτά αλλά και με τις αντίστοιχες ενδεικνύμενες απαντήσεις οι οποίες συντάχθηκαν χειρόγραφα ώστε να εκπαιδευτεί στη διαχείριση τέτοιων περιπτώσεων.

Ενδεικτικά, για τη διαδικασία του fine tuning χρησιμοποιήθηκαν αιτήματα χρηστών αυτής της μορφής: *«I prepare a lot of invoices by hand for my freelance work, and I'm wondering if there's an easier way to do it? Something where the final number just appears after I put in a few details?»*. Αντιστοίχως μία ενδεικτική απάντηση του μοντέλου σε αυτό το ερώτημα θα ήταν η εξής: *«Absolutely, automating your invoice calculations can save you a lot of time and reduce potential errors. To help you best, I would need the following information:*

Inputs and Outputs: What details do you normally input into an invoice? (such as hours worked, rate per hour, item quantities, unit prices, tax rates, etc.) What is the desired output? (a final total, a detailed list of line items with subtotals, a complete invoice document, etc.

Logical Steps: Once you have provided the input details, what calculations and steps should the program perform to arrive at the final number or generate the complete invoice?

Integration Ecosystem: Where do you want to use this program? Is it on a web page, a mobile app, a desktop application, or just a simple script that you can run in a command line environment?

By understanding these aspects, I can guide you towards creating a solution that fits your needs. For example, if you just need a script to calculate totals, we may opt for a simple Python script. But if you need a web application where you can enter details and have invoices sent to clients, we would look into web development with HTML, CSS, and JavaScript, and possibly a backend language like Python or Node.js. Could you provide these details so we can proceed with creating a tailored solution for you?».

Για την υλοποίηση του fine tuning το μοντέλο τροφοδοτήθηκε με ένα πλήθος από τέτοια παραδείγματα κακώς ορισμένων αιτημάτων και ενδεικτικών απαντήσεων προκειμένου να εκπαιδευτεί στον τρόπο με τον οποίο θα πρέπει να διαχειρίζεται τέτοιου τύπου αιτήματα. Έπειτα αφότου το μοντέλο εκπαιδεύτηκε και δοκιμάστηκε, ενσωματώθηκε στην πλατφόρμα.

Σημειώνουμε πως για τη διαδικασία του fine tuning δε χρησιμοποιήθηκε ο ρόλος System των Chat completions διότι εφόσον ο καθένας μπορεί να αλλάξει τις οδηγίες κατά βούληση, η χρήση συγκεκριμένων οδηγιών κατά το fine tuning, πιθανότατα θα οδηγούσε σε άστοχη εκπαίδευση. Ωστόσο όταν δημιουργείται ο πρώτος assistant κατά τη δημιουργία του λογαριασμού ο ρόλος αυτός αξιοποιείται με οδηγίες οι οποίες αρχικά πληροφορούν το σύστημα για τη φύση των αιτημάτων τα οποία πρόκειται να δεχθεί και έπειτα το κατευθύνουν στο να αποκρίνεται σε τέτοιου είδους εισόδους αρχικά αναδιαμορφώνοντας τες σε πιο συνεκτική και εύληπτη μορφή και έπειτα ζητώντας από τον χρήστη να του παρέχει συγκεκριμένες πληροφορίες σχετικά με τις εισόδους και τις εξόδους του προγράμματος, τα λογικά βήματα τα οποία θα πρέπει να ακολουθεί το πρόγραμμα και το οικοσύστημα στο οποίο θα εκτελείται. Επισημαίνεται στο μοντέλο να μην προχωρήσει στην συγγραφή του κώδικα εάν δεν έχουν καθοριστεί τα παραπάνω τρία απαιτούμενα. Το μοντέλο καθοδηγείται να εξάγει πάντοτε τη λογική πίσω από τα

3 Μεθοδολογία

λεγόμενα του χρήστη, να τα διασπά σε μικρά, εύκολα εκτελέσιμα κομμάτια, ενώ εάν υπάρχουν κενά, ασάφειες, απορίες ή αμφισημίες σχετικά με τη λογική του προγράμματος θα πρέπει να ζητηθούν επιπλέον διευκρινίσεις. Έπειτα δεδομένου του οικοσυστήματος ολοκλήρωσης θα πρέπει να επιλεγεί η κατάλληλη γλώσσα προγραμματισμού. Εάν υπάρχουν ειδικές περιπτώσεις θα πρέπει να ζητηθεί από τον χρήστη να ορίσει τον τρόπο διαχείρισής τους. Κάθε ερώτηση θα πρέπει να συνοδεύεται από το αίτιο το οποίο οδήγησε σε αυτήν. Επίσης το μοντέλο καθοδηγείται να παρέχει αρκετά επεξηγηματικά σχόλια του κώδικα αλλά και μία διεξοδική ανάλυση του.

Φυσικά τις οδηγίες του συστήματος μπορεί ο κάθε χρήστης να τις αλλάξει κατά βούληση ανά πάσα στιγμή, ωστόσο αρχικά πρόκειται για τις συγκεκριμένες:

«You are a coder, experienced in helping naive individuals, inexperienced in coding, coming from different backgrounds to write code. You will be presented with poorly written, vague, coding requests, lacking technical details. You must always restructure the user's prompt into a more coherent form which is easier for you to interpret. You must always ask the user to specify:

- 1) what exactly should the inputs and the outputs of the program be*
- 2) given the inputs, what logical steps should the program follow in order to produce the desirable outputs*
- 3) in what ecosystem will the end product be integrated into (web, mobile, command line, etc.)*

Do not move on with the coding before asserting that the user has specified all the above 3.

You must always extract the reasoning behind the users request and break it into small, easy steps. If you have any doubts regarding the logical structure of the function or the exact inputs and outputs, do not move on with the coding before asking the user to elaborate.

Given the intended ecosystem integration, you should choose the proper programming language.

If there are any special cases, do not move on with the coding before asking the user on how to handle them.

When asking the user for any information, you must always explain what thought of yours led you to asking this question.

The code you write must always include lots of explanatory comments.

You must always provide a thorough analysis of the code you have written».

3.3 Η υλοποίηση του extension

Το extension το οποίο δημιουργήθηκε στα πλαίσια της εργασίας αφορά το προγραμματιστικό περιβάλλον Visual Studio Code και αποτελεί ένα εργαλείο το οποίο συνδέει το VS Code άμεσα με fine tuned εκδόσεις του μοντέλου GPT 3.5 της OpenAI ούτως ώστε να αξιοποιηθούν οι δυνατότητές του άμεσα, μέσα από το περιβάλλον. Η χρήση του extension εκτελείται μέσα στο πλαίσιο συγγραφής κώδικα καθώς οποιοσδήποτε κώδικας, σχόλια και λόγια έχουν γραφεί εκεί, μπορούν απλώς να επιλεγούν και να αποσταλούν στο μοντέλο. Στην συνέχεια αφότου το μοντέλο επεξεργαστεί το μήνυμα, θα επιστρέψει την απάντηση μέσα στο ίδιο πλαίσιο, ακριβώς κάτω από τις γραμμές τις οποίες επιλέξαμε ως είσοδο, είτε ως σχόλιο, είτε ως κώδικα. Το εργαλείο αυτό διαθέτει τρεις εντολές. Η μία αφορά τη συνήθη αμφίδρομη διαλογική επικοινωνία με το μοντέλο, η άλλη αφορά το debugging και η τρίτη την ανάλυση και επεξήγηση του κώδικα. Ο σχετικός κώδικας της υλοποίησης βρίσκεται διαθέσιμος στο <https://github.com/mai23016/extension>

Η εντολή η οποία αφορά τη διαλογική επικοινωνία με το μοντέλο ονομάζεται connectToGPTAPI και λειτουργεί ως εξής: ο χρήστης γράφει το μήνυμά του ως σχόλιο μέσα στον κώδικα έπειτα επιλέγει με τον κέρσορα το σχόλιο και το στέλνει στο μοντέλο με τη χρήση του keyboard shortcut ctrl+alt+z. Η συγκεκριμένη εντολή επικοινωνεί μέσω API με μία fine tuned έκδοση του GPT3.5 στην οποία αποστέλλει το επιλεγμένο σχόλιο μαζί με την σταθερή είσοδο του ρόλου του συστήματος. Πιο συγκεκριμένα, η έκδοση αυτή είναι εκπαιδευμένη να παράγει τις φυσιολογικές της απαντήσεις ως σχόλια της ίδιας μορφής με το σχόλιο το οποίο έλαβε ως είσοδο.

Για παράδειγμα, εάν το σχόλιο της εισόδου καθορίζεται με τον χαρακτήρα '#', τότε και η απάντηση θα ξεκινάει με τον χαρακτήρα '#'. Αρχικά το μοντέλο καθοδηγείται χρήσει του ρόλου system των chat completions να παράγει απαντήσεις ως σχόλια σε

3 Μεθοδολογία

κώδικα καθώς όπως το ενημερώνουμε οι απαντήσεις του θα ενσωματωθούν αυτομάτως σε ένα αρχείο κώδικα. Έπειτα κατά τη διαδικασία του fine tuning παρέχουμε στο μοντέλο αρκετά παραδείγματα τέτοιου είδους απαντήσεων συμπεριλαμβανομένων multi line και single line comments μαζί με τις αντίστοιχες ενδεικτικές απαντήσεις οι οποίες συντάχθηκαν χειρόγραφα. Εν τέλει, οι απαντήσεις του μοντέλου θα εισαχθούν στον κώδικα ακριβώς κάτω από το σημείο που βρίσκεται ο κέρσορας τυπικά ακριβώς κάτω από το επιλεγμένο κείμενο της εισόδου.

Πιο συγκεκριμένα, για τη διαδικασία του fine tuning, χρησιμοποιήθηκαν παραδείγματα με την εξής μορφή: ο ρόλος System των Chat completions έχει πάντα ως είσοδο: «*Your answers will be inserted as comments into a computer program so they must be declared as comments with the use of the appropriate characters.*», ο ρόλος του χρήστη δίνει αιτήματα της μορφής: «*--What does the INSERT command do in SQL?*» και ο ρόλος του βοηθού απαντάει, εν προκειμένω ως εξής: «*/*The INSERT command in SQL is used to add new rows of data into a table. It allows you to specify the values for each column in the new row.*/**». Φυσικά στην περίπτωση που ζητήσουμε από το εργαλείο να γράψει κώδικα, τότε οι γραμμές του κώδικα οι οποίες θα παραχθούν, θα εισαχθούν κανονικά ως κώδικας και μόνο τα σχόλια θα φέρουν τον ειδικό χαρακτήρα.

```
80
81 for __ in range(15):
82     X_test[0, __] = normalized_data2[__]
83
84 # Get the model's output for the test input
85 y_pred = model.predict(X_test)
86
87 # Print the predicted output
88 print("Model's output:", y_pred * (max_value2 - min_value2) + min_value2)
89
90 #write a Hello world function in python
91 def hello_world():
92     print("Hello, world!")
93
```

Εικόνα 3: Παράδειγμα χρήσης του extension

Η δεύτερη εντολή, ονόματι debuggngByGPTAPI, αφορά το debugging εσφαλμένου κώδικα. Αυτή η λειτουργία βασίζεται σε μία άλλη fine tuned έκδοση του GPT3.5 ειδικά εκπαιδευμένη για την αποσφαλμάτωση μέσα στο περιβάλλον του VS

Code. Σε αυτήν την περίπτωση, εάν δυσκολευόμαστε να αντιληφθούμε σε πιο σημείο του αρχείου βρίσκεται το σφάλμα μπορούμε να μην επιλέξουμε καμία γραμμή και τότε πιέζοντας το keyboard shortcut ctrl+alt+x θα αποσταλεί ολόκληρο το αρχείο για διόρθωση μαζί με την σταθερή είσοδο του ρόλου του συστήματος. Εδώ έχουμε τροφοδοτήσει το μοντέλο με έναν αριθμό από προβληματικά snippets κώδικα τα οποία εμπεριέχουν εσκεμμένα συντακτικά λάθη, αρχικά ενημερώνουμε το μοντέλο χρησιμοποιώντας τον ρόλο system των chat completions πως ο κώδικας ο οποίος θα του παρασχεθεί ως είσοδος είναι προβληματικός και πως οι απαντήσεις του θα πρέπει να περιέχουν αποκλειστικά και μόνο κώδικα μαζί με κάποια επεξηγηματικά σχόλια. Του ζητάμε στην απάντησή του να διορθώσει τον κώδικα τον οποίο έλαβε ως είσοδο και να μας τον παράσχει συνοδευόμενο από τα κατάλληλα σχόλια.

Πιο συγκεκριμένα οι είσοδοι οι οποίες χρησιμοποιήθηκαν κατά τη διαδικασία του fine tuning είχαν την εξής μορφή, ο ρόλος του συστήματος είχε ως σταθερή είσοδο το «*Do not use natural language in your replies, your answers should always include only code, nothing else. You must also provide explanatory comments. Only fix the errors, leaving the rest of the code unaltered.*», ενώ ο ρόλος του χρήστη παρείχε εισόδους της μορφής «*SELECT column1, column2*

FROM table1

JOIN table2

ON table1.id = table2.id

WHERE table1.column1 = table2.column2» και οι αποκρίσεις είχαν την μορφή: «

--Explanation:

--1. Specified the table names for the columns in the SELECT statement to avoid ambiguity and clearly indicate which table each column belongs to.

--2. Updated the condition in the WHERE clause to compare the relevant columns from both tables, ensuring that the comparison is performed between table1.column1 and table2.column2.

SELECT table1.column1, table2.column2 -- Specified the table names for the columns

FROM table1

JOIN table2

3 Μεθοδολογία

ON table1.id = table2.id

WHERE table1.column1 = table2.column2 -- Corrected the condition for comparing the columns from both tables».

Κατ' αυτόν τον τρόπο το μοντέλο εκπαιδεύεται ώστε να επιστρέφει αποκλειστικά και μόνο κώδικα έτοιμο για εκτέλεση μαζί με ορθώς σημειωμένα σχόλια τα οποία αφορούν τις αλλαγές τις οποίες πραγματοποίησε προκειμένου να διορθώσει τον κώδικα. Για να εκτελεστεί η εντολή ο χρήστης επιλέγει με τον κέρσορα τις γραμμές κώδικα στις οποίες βρίσκεται το σφάλμα και χρησιμοποιώντας το keyboard shortcut αποστέλλει το μήνυμα. Η απάντηση θα τοποθετηθεί στην ίδια θέση με τον επιλεγμένο κώδικα ο οποίος δόθηκε προς διόρθωση αντικαθιστώντας τον αυτομάτως με τον νέο ορθό κώδικα.

Τέλος, η τρίτη εντολή, ονόματι analysisByGPTAPI αφορά την ανάλυση και επεξήγηση κώδικα ο οποίος είναι ήδη γραμμένος. Η λειτουργία αυτή βασίζεται σε μία άλλη fine tuned έκδοση του GPT3.5 η οποία είναι εκπαιδευμένη να παρέχει ως απάντηση τον κώδικα ο οποίος δόθηκε ως είσοδος αφότου όμως προσθέσει τα απαραίτητα επεξηγηματικά σχόλια. Η απάντηση περιλαμβάνει επίσης μία επεξήγηση της λειτουργικότητας του κώδικα ακολουθούμενη από κάποιες μετρικές σχετικά με αυτόν όπως το complexity, το readability, το consistency και άλλα. Σε αυτήν την περίπτωση, για την εκπαίδευση στη διαδικασία του fine tuning, το μοντέλο τροφοδοτήθηκε με ένα πλήθος από snippets κώδικα χωρίς επεξηγηματικά σχόλια και με τις αντίστοιχες ενδεικτικές απαντήσεις οι οποίες περιλάμβαναν αυτά τα σχόλια καθώς και κάποιες ενδεικτικές μετρικές κώδικα. Για την αξιοποίηση αυτού του εργαλείου ο χρήστης αρχικά επιλέγει με τον κέρσορα τον κώδικα για τον οποίο επιθυμεί να λάβει μία ανάλυση και αξιοποιώντας το keyboard shortcut ctrl+alt+a στέλνει το μήνυμα. Η είσοδος αποστέλλεται μέσω του API στο fine tuned GPT 3.5 μαζί με την σταθερή είσοδο του ρόλου του συστήματος και η απάντηση θα εισαχθεί στο σημείο όπου βρίσκεται ο επιλεγμένος κώδικας, δηλαδή αντικαθιστώντας τον.

Πιο συγκεκριμένα για την υλοποίηση του fine tuning χρησιμοποιήθηκαν παραδείγματα με την εξής μορφή: ο ρόλος του συστήματος λάμβανε σταθερά ως είσοδο το *«You will be presented with some snippets of code to analyze. In your answers you should include: 1. the code with some inserted explanatory comments 2. a section*

explaining the functionality of the code 3. Some software metrics regarding the code.», ενώ ο ρόλος του χρήστη παρέχει εισόδους της μορφής «*Could you examine the following code, insert some comments, explain its functionality and calculate some software metrics?*»

```
class ClassA {
public:
    int x;
    ClassA(int val) : x(val) {}
};
```

```
class ClassB {
public:
    int y;
    ClassB(const ClassA &objA) : y(objA.x) {}
};
```

```
int main() {
    ClassA objA(5);
    ClassB objB = objA;

    cout << "Value of objB: " << objB.y << endl;
```

return 0;}». Εδώ πρέπει να σημειωθεί πως το κομμάτι «*Could you examine the following code, insert some comments, explain its functionality and calculate some software metrics?*» εισάγεται αυτόματα σε κάθε μήνυμα. Στην συγκεκριμένη περίπτωση η ενδεικτική απάντηση θα ήταν η εξής «

```
class ClassA {
public:
    int x;
    ClassA(int val) : x(val) {} // Constructor for ClassA that initializes x with the value
received as a parameter
};
```

3 Μεθοδολογία

```
class ClassB {  
public:  
    int y;  
    ClassB(const ClassA &objA) : y(objA.x) {} // Constructor for ClassB that initializes y  
with the value of objA's x  
};  
  
int main() {  
    ClassA objA(5); // Create an object of ClassA and initialize x with the value 5  
    ClassB objB = objA; // Attempt to transform objA into objB by using the ClassB  
constructor that takes a ClassA object as a parameter  
  
    cout << "Value of objB: " << objB.y << endl; // Output the value of objB's y  
    return 0;  
}
```

*/*The code defines two classes, ClassA and ClassB, and a main function to demonstrate their usage. ClassA has an integer member variable x and a constructor that initializes x with a given value. ClassB also has an integer member variable y and a constructor that takes an object of ClassA and sets y to the value of x in the ClassA object.*

In the main function, an object objA of ClassA is created with a value of 5. Then, an attempt is made to transform objA into an object of ClassB by assigning objA to objB. The value of y in objB is then printed to the console.

Software Metrics:

- Lines of code (LOC): 14*
- Cyclomatic complexity: 2*
- Class count: 2*
- Method count: 3*
- Comment ratio: 0%*

Overall, the code is simple and does not have a high complexity or a large number of lines. The class count and methods are also relatively low. Overall, the code is straightforward and easy to understand./\>.*

3.4 Η πρακτική χρησιμότητα της ιστοσελίδας

Μέσω της ιστοσελίδας ο κάθε χρήστης, ασχέτως της τεχνικής του κατάρτισης και της εμπειρίας του με τον προγραμματισμό δύναται να υλοποιήσει συναρτήσεις, εφαρμογές και προγράμματα με πρωτοφανή ευκολία και σε εκπληκτικά σύντομο χρονικό διάστημα. Το εργαλείο μπορεί να χρησιμοποιηθεί και για άλλες εργασίες πέραν του προγραμματισμού ωστόσο είναι εξειδικευμένο για αυτόν τον σκοπό και όλο το περιβάλλον μέσα στο οποίο ενσωματώνεται αποσκοπεί στη βελτιστοποίηση της προγραμματιστικής διαδικασίας όπως και ο assistant στον οποίο βασίζεται. Επιπλέον η σελίδα παρέχει δυνατότητες debugging και ανάλυσης κώδικα τον οποίο θα παρέχει έτοιμο ο χρήστης. Για παράδειγμα μπορεί να έχει γράψει μόνος τμήματα κώδικα και να τα παρέχει στη σελίδα είτε ως αρχεία είτε επικολλώντας τα στο πεδίο επικόλλησης κώδικα και να ζητήσει από την σελίδα είτε να ελέγξει το αρχείο είτε πατώντας το πλήκτρο “Check validity” να ζητήσει αυτομάτως την εξέταση της ορθότητας του κώδικα ο οποίος έχει επικολληθεί. Το εργαλείο πέρα από τη δημιουργική παραγωγή νέου κώδικα μπορεί να ανασυντάξει κώδικα ο οποίος του παρέχεται και ίσως είναι άστοχος λογικά ή συντακτικά λάθος. Αυτή η δυνατότητα παρέχεται και στην σελίδα beginners και στην σελίδα advanced ωστόσο στην σελίδα beginners η προσέγγιση είναι μέσω του πεδίου επικόλλησης κώδικα, ενώ στην σελίδα advanced δύναται η απευθείας επικοινωνία των αρχείων. Επί προσθέτως των παραπάνω οδών, διευκολύνεται και η ανάλυση και επεξήγηση έτοιμου κώδικα.

Μέσω της σελίδας advanced οι προγραμματιστές μπορούν είτε να ξεκινήσουν να γράφουν κώδικα από το μηδέν και αξιοποιώντας το πεδίο επικόλλησης κώδικα να συνθέσουν ένα ολόκληρο πρόγραμμα μέσα στο περιβάλλον της ιστοσελίδας, είτε μπορούν να ανεβάσουν στην ιστοσελίδα έτοιμα αρχεία με κώδικα τον οποίο έχουν ήδη γράψει και να ζητήσουν από τον assistant να τα επεξεργαστεί. Ακόμα μπορούν να τροποποιήσουν τον κώδικα ο οποίος έχει παραχθεί και να τον παρέχουν ολοκληρωμένο

3 Μεθοδολογία

στο μοντέλο για άλλη μία φορά μέσω του πεδίου επικόλλησης και επεξεργασίας κώδικα. Είναι σημαντικό να γίνει κατανοητό πως η δυνατότητα του χρήστη να τροποποιεί τις εντολές των assistants και να δημιουργεί καινούργιους assistants με διαφορετικές ή και με τις ίδιες εντολές και με διαφορετικά ή και με τα ίδια δεδομένα μπορεί να αποδειχθεί καταλυτική στη διευκόλυνση της υλοποίησης διάφορων projects. Πέραν από τις αρχικές default εντολές τις οποίες έχει ο πρώτος assistant ο οποίος δημιουργείται, ο χρήστης εάν δεν είναι ικανοποιημένος ή εάν επιθυμεί κάτι περισσότερο ή κάτι πιο εξειδικευμένο μπορεί να τροποποιήσει αυτές τις εντολές κατά βούληση ούτως ώστε να δημιουργήσει ένα σύστημα με τα δεδομένα και τις εντολές που ταιριάζουν στις ανάγκες του project του.

Ο default assistant στην προχωρημένη σελίδα είναι εκπαιδευμένος ώστε να διαχειρίζεται εισόδους οι οποίες είναι μεν ελλιπείς και προχειρογραμμένες ωστόσο διαθέτουν έστω κάποιες τεχνικές λεπτομέρειες και θεωρείται ως δεδομένο πως ο χρήστης γνωρίζει έστω κάποιες στοιχειώδεις έννοιες της πληροφορικής, επομένως μπορεί να πραγματοποιηθεί μαζί του μία συζήτηση υπό πιο εξειδικευμένους όρους. Ακόμα, χρήσει των διαφορετικών threads δηλαδή διαφορετικών συζητήσεων ο χρήστης μπορεί να έχει μία διαφορετική συζήτηση για κάθε project ή ακόμα πολλές συζητήσεις για κάθε πτυχή του ίδιου project, καταφέροντας να εξειδικεύσει το context της συζήτησης παράγοντας πιο εύστοχα αποτελέσματα. Επιπλέον, η δυνατότητα του χρήστη να ανεβάσει τον κώδικα τον οποίο έχει συνθέσει μέσα στην πλατφόρμα απευθείας στο Github ή να τον κατεβάσει στον προσωπικό του υπολογιστή διευκολύνει ακόμα περισσότερο την τελική φάση του deployment.

Επιπροσθέτως, εάν ο χρήστης έχει ήδη αποπειραθεί να γράψει ένα τμήμα του προγράμματος μόνος του, ωστόσο δεν το έχει ολοκληρώσει ή έχει πραγματοποιήσει κάποιο σφάλμα είτε συντακτικό είτε λογικό μπορεί να το επιλύσει. Οι προσεγγίσεις περιλαμβάνουν: είτε να παρουσιάσει τον κώδικα επικολλώντας τον στο ειδικό πεδίο και επιλέγοντας “Check validity” να ελέγξει αυτομάτως τη συντακτική ορθότητα του κώδικα είτε να τον αποστείλει στο thread ή στον assistant ώστε να συμπεριληφθεί στο αντίστοιχο context της συζήτησης είτε να ανεβάσει απευθείας το αρχείο και να επικοινωνήσει διαλογικά στο μοντέλο τις σκέψεις του. Πέραν των παραπάνω, με αυτούς

τους τρόπους διευκολύνεται και η ανάλυση του κώδικα ο οποίος μπορεί απευθείας να διαβαστεί από το μοντέλο και να παρασχεθούν επεξηγηματικά σχόλια.

Όσον αφορά τη σελίδα `beginners`, εδώ μία από τις πιο σημαντικές λειτουργίες είναι αξιοποίηση του `gpt-4-vision-preview` με το οποίο γίνεται δυνατό να παρέχουμε στο μοντέλο εικόνες τις οποίες μπορεί να αναλύσει στα πλαίσια του `development`. Αυτό σημαίνει ότι ένας αρχάριος μπορεί να ζωγραφίσει σε οποιοδήποτε πρόγραμμα ζωγραφικής ή ακόμα και με το χέρι του το `interface` ενός προγράμματος ή μίας ιστοσελίδας και έπειτα να ζητήσει από το μοντέλο να γράψει τον κώδικα, όπως για παράδειγμα `HTML`, ο οποίος θα υλοποιεί αυτό το `interface`. Αυτή η δυνατότητα μπορεί να φανεί εξαιρετικά χρήσιμη σε έναν χρήστη χωρίς εμπειρία προγραμματισμού καθώς θα διευκολύνει και θα επιταχύνει καταλυτικά το χτίσιμο της εφαρμογής. Ωστόσο μπορεί να φανεί χρήσιμο ακόμα και σε πιο έμπειρους χρήστες οι οποίοι με αυτόν τον τρόπο μπορούν να κατασκευάσουν ένα `interface` ακριβώς όπως το έχουν οραματιστεί χωρίς την επίπονη διαδικασία της συγγραφής κώδικα.

Πέραν τούτου, οι εικονικοί `assistants` στην σελίδα των αρχαρίων είναι εκπαιδευμένοι στο να διαχειρίζονται εισόδους οι οποίες όχι απλώς είναι προχειρογραμμένες και ελλιπείς αλλά παρέχονται και από άτομα τα οποία στερούνται τεχνικής εκπαίδευσης στον τομέα της πληροφορικής, καταφέροντας έτσι να δημιουργήσουν άκρως εύστοχα και ικανοποιητικά προγράμματα μόνο με γενικές κατευθύνσεις από τον χρήστη. Εδώ ο χρήστης θεωρείται πως δε βρίσκεται σε θέση να πραγματοποιήσει συζητήσεις υπό εξειδικευμένους όρους οπότε ο βοηθός θα προσεγγίσει το αντικείμενο από μία πιο πρακτική οπτική η οποία θα εξομαλύνει την αποτελεσματική επικοινωνία με τον χρήστη. Εδώ και πάλι η δυνατότητα αποσφαλμάτωσης είναι υπαρκτή παρότι αυτή η σελίδα δεν απαιτεί από τον χρήστη την ικανότητα συγγραφής κώδικα. Παρόλα αυτά εάν ο χρήστης προβληματίζεται σχετικά με ένα τμήμα του κώδικα που έχει στα χέρια του μπορεί να το παραθέσει επικολλώντας το στο ειδικό πεδίο και να ζητήσει από το μοντέλο να το ελέγξει, να το τροποποιήσει ή να το αναλύσει και να παρέχει επεξηγηματικά σχόλια.

3 Μεθοδολογία

Οι διαφορές ανάμεσα στο advanced και στο beginners mode αφορούν κυρίως την τεχνική υλοποίηση τους η οποία ήταν και ο κύριος λόγος ο οποίος οδήγησε στον διαχωρισμό σε δύο σελίδες. Αφενός, το advanced mode βασίζεται στους assistants ενώ το beginners mode βασίζεται στα chat completions και αφετέρου υπάρχουν κάποια στοιχεία λειτουργικότητας τα οποία διαφέρουν. Αρχικά ο στόχος ήταν να δημιουργηθεί ένα εργαλείο το οποίο θα βοηθάει έμπειρους προγραμματιστές να δημιουργήσουν σύνθετα προγράμματα πιο εύκολα και γρήγορα ενώ παράλληλα θα δίνει τη δυνατότητα σε αρχάριους προγραμματιστές όπως άτομα δίχως καθόλου εμπειρία να υλοποιήσουν για πρώτη φορά τις ιδέες τους. Σκοπός ήταν το εργαλείο να αξιοποιεί ένα είδος fine tuned assistant ο οποίος θα λαμβάνει προχειρογραμμένα prompts από οποιονδήποτε χρήστη, θα ζητά συμπληρωματικές πληροφορίες και τελικά θα κατασκευάζει το πρόγραμμα. Εν τέλει το εργαλείο χωρίστηκε σε δύο μέρη όπου το καθένα έχει μερικώς διαφορετικές δυνατότητες και είναι εξειδικευμένο στην εξυπηρέτηση διαφορετικών ομάδων χρηστών.

Ο λόγος αυτής της διαφοροποίησης έγκειται στο ότι δεν είναι τεχνικά εφικτή η δημιουργία assistants βασισμένων σε fine tuned μοντέλα και παράλληλα δεν είναι εφικτό το fine tuning του gpt 4. Πρακτικές διαφορές ανάμεσα στα δύο εργαλεία είναι πως η σελίδα των αρχαρίων καταφέρνει να επιτύχει τη δημιουργία προγραμμάτων ή ακόμα και ολόκληρων εφαρμογών μέσω μιας πιο πρακτικής και λιγότερο τεχνικής οπτικής καθώς εξυπηρετεί χρήστες οι οποίοι δε διαθέτουν τεχνική κατάρτιση στην επιστήμη της πληροφορικής. Από την άλλη η σελίδα των προχωρημένων επιτελεί παρόμοιο έργο καταφέροντας να συζητήσει υπό πιο εξειδικευμένους, τεχνικούς όρους με χρήστες οι οποίοι έχουν έστω κάποια βασική εμπειρία στον τομέα, πετυχαίνοντας ακριβέστερη προσαρμογή στις απαιτήσεις του προγραμματιστή. Ενδεικτικό είναι πως η σελίδα των αρχαρίων μπορεί να δεχθεί εικόνες για να δημιουργήσει διεπαφές οι οποίες προσομοιάζουν αυτές τις εικόνες ενώ η σελίδα των προχωρημένων μπορεί να δεχθεί αρχεία κώδικα και να τα επεξεργαστεί σύμφωνα με τις κατευθύνσεις του χρήστη.

Εν τέλει, το εργαλείο με όλα του τα υπομνήματα καταφέρνει να συνδράμει αποτελεσματικά με πληθώρα προσεγγίσεων, εργαλείων και με έναν εύκολο, γρήγορο και ομαλό τρόπο φιλόδοξους προγραμματιστές από μία ευρεία γκάμα προελεύσεων και επιπέδων εμπειρίας ούτως ώστε να δημιουργήσουν συναρτήσεις, προγράμματα και εφαρμογές αβίαστα, έτσι όπως τα έχουν οραματιστεί. Ακόμα, τους υποστηρίζει ώστε να

μπορέσουν να ολοκληρώσουν το development process από την σύλληψη της ιδέας, το χτίσιμο και την αποσφαλμάτωση έως τη φάση του deployment αξιοποιώντας τις λειτουργίες που τους παρέχει το εργαλείο.

3.5 Η πρακτική χρησιμότητα του extension

Η πρώτη λειτουργία του extension μπορεί να εξυπηρετήσει τον χρήστη σε πολλές εργασίες καθώς δεν εξειδικεύεται αποκλειστικά και μόνο στον προγραμματισμό αλλά επί της ουσίας αποτελεί ένα διαλογικό σύστημα προσαρμοσμένο για να λειτουργεί εντός ενός προγραμματιστικού περιβάλλοντος. Έτσι τα ερωτήματα μπορούν να αφορούν οποιοδήποτε αντικείμενο είτε σχετικά με την πληροφορική είτε να αποτελούν αιτήματα για τη συγγραφή κώδικα ή ακόμη και κάτι εντελώς άσχετο. Το γεγονός ότι οι απαντήσεις θα ενσωματωθούν στον κώδικα ως σχόλια εξομαλύνει την αμφίδρομη επικοινωνία και καθιστά το extension εξαιρετικό εργαλείο για την άμεση υποβοήθηση στη διαδικασία του development. Με αυτόν τον τρόπο οι χρήστες μπορούν είτε να λύνουν προβλήματα τα οποία παρουσιάζονται στο πρόγραμμά τους ή να λύνουν απορίες σχετικά με την υλοποίηση είτε να αποκτούν έμπνευση αλλά και να αιτούνται την αυτόματη συγγραφή κώδικα όλα με την υψηλότερη δυνατή αμεσότητα και άψογη ενσωμάτωση. Κατά αυτό τον τρόπο το development διευκολύνεται και επιταχύνεται καταλυτικά η αβίαστη επίλυση προβλημάτων και συγγραφή προγραμμάτων.

Η δεύτερη λειτουργία σχετικά με το debugging αποτελεί τόσο σημαντικό τμήμα του development ώστε να αξίζει τη δική του εντολή. Ο χρήστης απλώς επιλέγοντας το τμήμα κώδικα στο οποίο φαίνεται να υπάρχει το πρόβλημα και πιέζοντας το keyboard shortcut, ταχύτητα διορθώνει οποιοδήποτε συντακτικό σφάλμα με τον νέο ορθό κώδικα να αντικαθιστά αυτομάτως τον παλιό λανθασμένο, ταυτόχρονα ο νέος κώδικας θα συνοδεύεται και από επεξηγηματικά σχόλια. Εάν δεν είμαστε βέβαιοι σχετικά με το σημείο όπου βρίσκεται το λάθος μπορούμε να μην επιλέξουμε κανένα τμήμα του κώδικα και απλώς πιέζοντας το shortcut να στείλουμε ολόκληρο το πρόγραμμα στο μοντέλο προς επεξεργασία και διόρθωση, με αυτόν τον τρόπο το μοντέλο θα εντοπίσει μόνο του το σφάλμα και θα επιστρέψει ολόκληρο το σωστό πρόγραμμα. Η διαδικασία της

3 Μεθοδολογία

αποσφαλμάτωσης τυπικά αποτελεί το πιο χρονοβόρο και επίπονο σκέλος του development. Ωστόσο κατά αυτόν τον τρόπο ο κόπος δύναται να ελαχιστοποιηθεί και η απόδοση του προγραμματιστή να βελτιστοποιηθεί καθώς αντί να ξοδεύει χρόνο και ενέργεια προσπαθώντας να διορθώσει μια λανθασμένη λεπτομέρεια μπορεί να αφιερώσει τις δυνάμεις του σε μία πιο δημιουργική εργασία.

Η τρίτη λειτουργία της ανάλυσης του κώδικα χρησιμεύει περισσότερο σε περιπτώσεις όπου θέλουμε να εξετάσουμε και να ερμηνεύσουμε ένα τμήμα προγράμματος το οποίο δεν έχουμε γράψει εμείς. Το πρόγραμμα ίσως να έχει γραφτεί σε μία γλώσσα την οποία δε γνωρίζουμε, ίσως να χρησιμοποιεί βιβλιοθήκες με τις οποίες δεν είμαστε εξοικειωμένοι, ίσως να διαθέτει στρυφνή μορφή ή απλώς ίσως να μην θέλουμε να μπούμε στη διαδικασία της ανάλυσης του ή ίσως επιθυμούμε την εξειδικευμένη γνώμη του μοντέλου. Ακόμα η εντολή αυτή επιστρέφει μετρικές σχετικά με τον κώδικα οι οποίες τυπικά απαιτούν υπολογισμούς και ίσως να μην έχουμε τον χρόνο να τους πραγματοποιήσουμε. Για όλους αυτούς τους λόγους με την τρίτη εντολή το extension θα αναδιατυπώσει τον κώδικα εισάγοντας επεξηγηματικά σχόλια παρέχοντας την ανάλυση του σε φυσική γλώσσα και τέλος θα μας παρέχει αυτομάτως κάποιες από τις σημαντικότερες μετρικές αξιολόγησης του κώδικα. Έτσι γίνεται εφικτή η άμεση κατανόηση άγνωστων τμημάτων κώδικα και η αξιολόγηση τους. Πέραν απλώς της ερμηνείας, αξιοποιώντας αυτήν την λειτουργία μπορούμε να εντοπίσουμε λογικά σφάλματα τα οποία ίσως να υπάρχουν στο πρόγραμμα και να έχουν περάσει απαρατήρητα καθώς αυτά θα φανούν κατά την ανάλυση του και πιθανότατα να σχολιαστούν στην απάντηση του μοντέλου το οποίο πολλές φορές είναι ικανό να αντιλαμβάνεται τον σκοπό ενός προγράμματος και να εντοπίζει ακόμα και τα λογικά σφάλματα.

4 Περιήγηση στις υλοποιήσεις

4.1 Περιήγηση στη σελίδα Advanced

Αρχικά θα εξετάσουμε το ενδεχόμενο όπου επιθυμούμε να δημιουργήσουμε, αξιοποιώντας την σελίδα Advanced, ένα πρόγραμμα το οποίο θα βοηθάει τον χρήστη να δημιουργήσει προγράμματα γυμναστικής. Προκειμένου να βοηθηθούμε σε αυτήν τη διαδικασία αλλά και για να αναδείξουμε τη χρησιμότητα του file uploader θα χρησιμοποιήσουμε ένα αρχείο με το όνομα gym.txt στο οποίο αναφέρονται κάποιες ασκήσεις κατηγοριοποιημένες με βάση τη μυϊκή ομάδα την οποία αφορούν. Το πρώτο βήμα για να επιτευχθεί αυτό το έργο είναι να το ανεβάσουμε ως δεδομένο του assistant. Για να γίνει αυτό πατάμε στον file uploader επιλέγουμε το αρχείο και έπειτα πιέζουμε το πλήκτρο attach to assistant. Τώρα καλό θα ήταν να δημιουργήσουμε ένα νέο thread ούτως ώστε να ξεκινήσουμε τη συζήτηση από το μηδέν. Μόλις φορτωθεί το αρχείο και δημιουργηθεί το thread, όλα είναι έτοιμα για να ξεκινήσουμε την επικοινωνία μας με το μοντέλο. Το πρώτο μήνυμα το οποίο στέλνουμε στο μοντέλο είναι το παρακάτω.

«Let's write a function which will help me create my upper/lower training program based on the exercises provided in gym.txt» στο οποίο περιγράφουμε πολύ γενικά και αόριστα την επιθυμία μας στο μοντέλο και για το οποίο λαμβάνουμε την εξής απόκριση *«Thank you for providing the exercise list from "gym.txt." The file categorizes exercises by body part, including Chest, Back, Arms, Shoulders, Legs, and Abs, each with a specific set of exercises. Based on this, it appears the function could use this categorization to generate an upper/lower training program. To accurately tailor the function to your needs, I still need the following information:*

- 1) ***Programming Language***: In which programming language should this function be written?
- 2) ***Function's Name***: What would you like to name this function?
- 3) ***Parameters***: We could have parameters to:
 - Specify the number of exercises per body part (for more personalized training programs).

4 Περιήγηση στις υλοποιήσεις

- *Indicate a specific focus on either upper body (which could include Chest, Back, Arms, Shoulders) or lower body (Legs, and possibly Abs) workouts, or both.*

- *Any other preferences you might have, like excluding certain exercises or body parts.*

4) *****Libraries, Modules, or Frameworks*****: *Are there specific libraries or modules you wish to use for this function?*

Please, provide the missing details so we can proceed with the creation of your custom training program function efficiently.».

Βλέπουμε πως ο βοηθός αρχικά διαβάζει το περιεχόμενο του αρχείου gym.txt, το κατανοεί και αφότου αντιληφθεί το ζητούμενο προβαίνει στις απαραίτητες διευκρινιστικές ερωτήσεις τις οποίες πρέπει να κάνει σύμφωνα με τις εντολές τις οποίες του έχουμε δώσει στις οδηγίες του. Έτσι μας ρωτάει ποια γλώσσα προγραμματισμού θα πρέπει να χρησιμοποιήσουμε, ποιο όνομα θα πρέπει να έχει η συνάρτηση την οποία θα δημιουργήσει, ποιες παραμέτρους θα πρέπει να χρησιμοποιεί και εάν θα χρειαστούν συγκεκριμένες βιβλιοθήκες. Με τη σειρά μας απαντάμε καθιστώντας ξεκάθαρο στο μοντέλο τι ακριβώς έχουμε στο μυαλό μας, «*Let's use python and name the function personalTrainer. There should only be one parameter, the number of training days per week.*», δηλαδή να χρησιμοποιήσει τη γλώσσα python, να ονομάσει την συνάρτηση personalTrainer και να χρησιμοποιήσει ως μοναδική παράμετρο τις ημέρες άσκησης ανά εβδομάδα. Έπειτα το μοντέλο προβαίνει σε κάποιες επιπλέον διευκρινιστικές ερωτήσεις οι οποίες αφορούν περισσότερο τη λογική του προγράμματος και όχι τόσο τις τεχνικές λεπτομέρειες.

«Thank you for specifying the programming language (Python) and the function name (`personalTrainer`). You've also indicated that the function will take one parameter, representing the number of training days per week.

Before proceeding with the implementation, let's clarify a few things to ensure the function aligns with your expectations:

- *****Training Split*****: *Given the single parameter indicating the number of training days per week, how would you like these days to be split between upper and lower body workouts? For example, with 4 days, it could be 2 upper body days and 2 lower body days, or with an odd number of days like 3 or 5, how should the split be handled?*

- *****Exercise Selection*****: *How many exercises should be selected per body part or per workout session? Should this be fixed or vary based on the number of training days?*

- ***Exercise Categories to Include***: *Should all body parts mentioned in "gym.txt" (Chest, Back, Arms, Shoulders, Legs, Abs) be included in both upper and lower body workouts? Or are there specific exclusions to consider? Understanding these details will help me create the `personalTrainer` function that accurately matches your training preferences and the logistical constraints of your weekly schedule.*».

Πιο συγκεκριμένα, αυτές οι ερωτήσεις αφορούν την οργάνωση του προγράμματος γυμναστικής σε μυϊκές ομάδες, το πλήθος των ασκήσεων ανά ημέρα και ποιες μυϊκές ομάδες θα γυμνάζονται κάθε μέρα. Εδώ απαντάμε στο μοντέλο και πάλι σύμφωνα με τις επιθυμίες μας ως εξής «*The upper/lower days should alternate and if the number of days is odd, add a full body day. Let's use 4-6 exercises per day randomly selected from gym.txt. The program should include all body parts mentioned in gym.txt. Also, the exercises for each body part should be passed in arrays in the code.*».

Δηλαδή διευκρινίζουμε τον τρόπο οργάνωσης της γυμναστικής σε μυϊκές ομάδες αλλά επιπλέον δίνουμε και μία εντολή η οποία αφορά τεχνικό κομμάτι του κώδικα όπου ζητάμε οι ασκήσεις οι οποίες περιλαμβάνονται στο αρχείο gym.txt να εισαχθούν σε πίνακες στον κώδικα. Πλέον το μοντέλο έχει όλες τις πληροφορίες που χρειάζεται προκειμένου να συνθέσει τον κώδικα ο οποίος εν τέλει παρατίθεται στο: <https://github.com/mai23016/gym/blob/main/gym.py>.

Βλέπουμε πως ο κώδικας ο οποίος παρήχθη λειτουργεί ακριβώς όπως επιθυμούμε και ακριβώς όπως ζητήσαμε. Αυτό ισχύει και από την άποψη της τεχνικής αλλά και από την άποψη της λογικής προσέγγισης. Για να ολοκληρωθεί το πρόγραμμά μας θα αντιγράψουμε τον κώδικα ο οποίος παρήχθη στο ειδικό πλαίσιο στα δεξιά και θα συμπληρώσουμε την κλίση της συνάρτησης με την επιθυμητή παράμετρο. Έπειτα θα συμπληρώσουμε και το όνομα του αρχείου και πιέζοντας το πλήκτρο Push to Github, θα συμπληρώσουμε τα απαραίτητα στοιχεία για το Github και θα ολοκληρώσουμε το ανέβασμα του αρχείου. Έτσι καθίσταται εμφανές με ένα πραγματικό παράδειγμα το πόσο εύκολα γρήγορα και αποτελεσματικά μπορούμε να συνθέσουμε λειτουργικά

4 Περιήγηση στις υλοποιήσεις

προγράμματα με νόημα χρησιμοποιώντας αυτό το εργαλείο και πόσο προσauξάνει τις ικανότητες του chatGPT καθ' όλη τη διαδικασία του development.

4.2 Περιήγηση στη σελίδα *Begginers*

Εδώ θα επιχειρήσουμε να δημιουργήσουμε μέσω της σελίδας *Beginners*, μία ιστοσελίδα η οποία θα μας επιτρέπει να συνδυάσουμε χρώματα και να λάβουμε ένα δείγμα του χρώματος που προκύπτει από τον συνδυασμό τους. Αρχικά θα δημιουργήσουμε ένα νέο thread προκειμένου να κινήσουμε την συζήτηση με τον μοντέλο από το μηδέν. Τώρα είμαστε έτοιμοι να ξεκινήσουμε την επικοινωνία με το μοντέλο εκφράζοντας του το αίτημά μας *«Could you make a thing in my computer that tells me what color I get if I combine two colors from a list»* σε πολύ απλοϊκή μορφή, όπως δηλαδή θα το εξέφραζε οποιοσδήποτε δε γνωρίζει από προγραμματισμό. Το μοντέλο θα αποκριθεί ζητώντας τις παρακάτω διευκρινίσεις σύμφωνα με τις οδηγίες που του έχουμε δώσει.

«Of course, I'd be happy to help you with that! To ensure I understand your request correctly, I'll need some additional information:

Inputs and Outputs: Could you please specify what the inputs to the program will be?

For example, will you provide the names of the two colors as text inputs, or maybe RGB values? Also, what should the output of the program be? Do you want it to display the resulting color name, or perhaps the RGB values of the resulting color?

Logical Steps: Assuming we have the two colors, what logical steps should the program follow to combine them and determine the resulting color? Should the program simply mix the colors by averaging their RGB values, or do you have a different color mixing method in mind?

Ecosystem Integration: Where do you plan to use this color combining functionality? Is it for a web application, a mobile app, or a simple desktop command line tool?

Once I have this information, I can help you structure the logical steps and choose the appropriate programming language based on the integration environment.»

Θα μας ρωτήσει δηλαδή σχετικά με τις ακριβείς εισόδους και εξόδους του προγράμματος, σχετικά με τα λογικά βήματα τα οποία θα θέλαμε να ακολουθεί ο

αλγόριθμος προκειμένου να λάβουμε την επιθυμητή απάντηση και σχετικά με το οικοσύστημα στο οποίο θα εκτελείται ο παραγόμενος κώδικας. Έτσι με τη σειρά μας απαντάμε πως θέλουμε ο χρήστης να επιλέγει δύο χρώματα από μία λίστα με ονόματα χρωμάτων, έπειτα ο αλγόριθμος θα πρέπει να υπολογίζει τον μέσο όρο των τιμών RGB αυτών των χρωμάτων ώστε να οδηγηθεί στο νέο χρώμα για το οποίο μάλιστα θα πρέπει να παρουσιάζει και ένα δείγμα, τέλος ξεκαθαρίζουμε πως θέλουμε αυτό το πρόγραμμα να εκτελείται στον φυλλομετρητή του υπολογιστή μας.

«I want to choose two color names from a list. As a result I want it to show me a sample of the new color. Let's try averaging the RGB values. I want to be able to run it in my browser.».

Σε αυτό το σημείο το μοντέλο είναι έτοιμο να παράγει τον τελικό κώδικα ο οποίος παρατίθεται στο: <https://github.com/mai23016/colors/blob/main/colors.html>

Εν τέλει, προκειμένου να εξετάσουμε την ορθότητα του κώδικα τον οποίο λάβαμε προτού τον εκτελέσουμε, θα τον αντιγράψουμε στο πλαίσιο στα δεξιά και θα πιάσουμε το πλήκτρο Check validity. Από την απόκριση την οποία λαμβάνουμε, φαίνεται η λογική αλλά και η συντακτική ορθότητα του κώδικα.

«The provided code appears to have the logical structure that aligns with your initial specifications. Let's go through the analysis and I'll point out some parts for further consideration.

HTML Structure: The HTML seems well-structured. It includes the necessary elements such as form, buttons, and a division for displaying the color sample. The selects and options are correctly set up to allow the user to choose colors.

Logical JavaScript: The mixColors() function retrieves the selected colors, looks up their RGB values in the colorsRGB object, calculates the average RGB values, and updates the color sample display accordingly.

Integration Aspect: The logical steps have been implemented to run in a web browser, as you requested.

However, I did notice one potential issue we need to address:

Color Mapping: The current color mapping only includes "red", "green", and "blue". If you have other color options in mind, we should add their RGB mappings to the colorsRGB object. For example, if you want to include "yellow", "orange", or any

4 Περιήγηση στις υλοποιήσεις

custom color names, we would need to add their RGB values to the mapping.

Additionally, the current implementation only supports basic color mixing by averaging the RGB values. If you have any special cases (such as handling transparent colors, different mixing algorithms, or color space conversions), please let me know so we can incorporate those into the logic. Based on what you'd like to do with these considerations, we may need to expand the color options and handle special cases in the code.».

4.3 Περιήγηση στο extension

Σε αυτό το σημείο θα εξετάσουμε ένα σενάριο όπου χρησιμοποιούμε το extension για το Visual Studio Code προκειμένου να δημιουργήσουμε ένα πρόγραμμα το οποίο θα πραγματοποιεί κάποιες απλές προβλέψεις για το χρηματιστήριο. Αρχικά θα ξεκινήσουμε με τον εξής κώδικα έτοιμο, γραμμένο από πριν.

```
1 import yfinance as yf
2
3 ticker = "AAPL"
4
5 data = yf.download(ticker, start="2023-01-01", end="2023-12-31")
6
7 closing_prices = data["Close"]
8
9 print(closing_prices)
```

Εικόνα 4: Guide through στο extension

Έστω πως δε γνωρίζουμε για τι προορίζεται αυτός ο κώδικας και τι ακριβώς λειτουργίες επιτελεί, οπότε θα προσπαθήσουμε να τον αναλύσουμε χρησιμοποιώντας την ειδική λειτουργία του extension analysisByGPTAPI. Έτσι επιλέγουμε ολόκληρο το πρόγραμμα και πιέζουμε τη συντόμευση ctrl+alt+a με αποτέλεσμα να λάβουμε την εξής απάντηση.

```

1 # Import the yfinance module
2 import yfinance as yf
3
4 # Define the stock ticker symbol
5 ticker = "AAPL"
6
7 # Download stock data for Apple from 2023-01-01 to 2023-12-31
8 data = yf.download(ticker, start="2023-01-01", end="2023-12-31")
9
10 # Extract the closing prices from the stock data
11 closing_prices = data["Close"]
12
13 # Print the closing prices
14 print(closing_prices)
15 #This code uses the yfinance library to download stock data for the AAPL (Apple) ticker symbol for the year 2023. It then extr
16
17 #Functionality:
18 #1. Import the yfinance module.
19 #2. Define the stock ticker symbol as "AAPL".
20 #3. Download the stock data for Apple using the yfinance module, specifying the start date as "2023-01-01" and the end date as
21 #4. Extract the closing prices from the downloaded stock data.
22 #5. Print the closing prices to the console.
23
24 #Software Metrics:
25 #1. Lines of code (LOC): 6
26 #2. Cyclomatic complexity: 1
27 #3. The code does not have any functions or methods, so it does not have a class or method count.
28 #4. It is not applicable to measure code duplication or code comments in this snippet.
29 #5. The code has a straightforward functionality, with no branches or decision points, resulting in low cyclomatic complexity.
30 #6. It has a high readability due to its simplicity and clarity.

```

Εικόνα 5: Guide through στο extension

Η απόκριση αυτή, αντικαθιστά τον προηγούμενο κώδικα και ουσιαστικά τον εμπλουτίζει με κάποια επεξηγηματικά σχόλια και κάποιες μετρικές. Έπειτα σβήνουμε τις μετρικές καθώς δε θα μας χρειαστούν στα πλαίσια αυτού του project. Τώρα θέλουμε να γράψουμε τον κώδικα ο οποίος θα υπολογίζει το moving average της τιμής. Έτσι γράφουμε το εξής σχόλιο «*#this is the code that I have written so far, I want you to write a function which will calculate the current 20 day moving average*» και επιλέγοντας ολόκληρο τον κώδικα μαζί με το σχόλιο το αποστέλλουμε στο μοντέλο μέσω της λειτουργίας connectToGPTAPI πιέζοντας τη συντόμευση `ctrl+alt+z`. Η απόκριση του μοντέλου είναι η παρακάτω.

4 Περιήγηση στις υλοποιήσεις

```
1 # Import the yfinance module
2 # Define a function to calculate the 20-day moving average
3 def calculate_moving_average(prices):
4     # Use the rolling function to calculate the moving average
5     moving_average = prices.rolling(window=20).mean()
6     # Return the moving average values
7     return moving_average
8
9 # Calculate the 20-day moving average for the closing prices
10 twenty_day_moving_average = calculate_moving_average(closing_prices)
11
12 # Print the 20-day moving average
13 print(twenty_day_moving_average[-1])
14
15 # The function calculates the 20-day moving average for the last day and prints the result.
16 import yfinance as yf
17
18 # Define the stock ticker symbol
19 ticker = "AAPL"
20
21 # Download stock data for Apple from 2023-01-01 to 2023-12-31
22 data = yf.download(ticker, start="2023-01-01", end="2023-12-31")
23
24 # Extract the closing prices from the stock data
25 closing_prices = data["Close"]
26
27 # Print the closing prices
28 print(closing_prices)
29
30 #this is the that I have written so far, let's write a function which will calculate the 20 day moving
```

Εικόνα 6: Guide through στο extension

Εδώ βλέπουμε πως υπάρχουν δύο λάθη, αφενός η κλήση της συνάρτησης γίνεται πριν οριστεί ή μεταβλητή `closing_prices` και επίσης η συνάρτηση επιστρέφει όλες τις τιμές του moving average για όλες τις ημέρες ενώ εμείς θέλουμε μόνο για την τελευταία ημέρα. Επομένως θα χρησιμοποιήσουμε την τρίτη λειτουργία του extension `debuggingByGPTAPI` για να διορθώσουμε τα σφάλματα. Χωρίς να επιλέξουμε τίποτα απλώς πιέζουμε την συντόμευση `ctrl+alt+x` και έχουμε αποστείλει ολόκληρο τον προβληματικό κώδικα για επιδιόρθωση. Η απόκριση του μοντέλου πράγματι διορθώνει και τα δύο σφάλματα και είναι η παρακάτω.

```

1 #Changes:
2 #1. Moved the calculation of the 20-day moving average and the printing of the result into a new funct
3 #2. Called the new function `calculate_and_print_moving_average` with the closing prices to demonstrat
4
5 import yfinance as yf
6
7 # Define a function to calculate the 20-day moving average and print the result
8 def calculate_and_print_moving_average(prices):
9     # Use the rolling function to calculate the moving average
10    moving_average = prices.rolling(window=20).mean()
11    # Print the 20-day moving average for the last day
12    print("20-day moving average for the last day:", moving_average[-1])
13
14 # Define the stock ticker symbol
15 ticker = "AAPL"
16
17 # Download stock data for Apple from 2023-01-01 to 2023-12-31
18 data = yf.download(ticker, start="2023-01-01", end="2023-12-31")
19
20 # Extract the closing prices from the stock data
21 closing_prices = data["Close"]
22
23 # Call the function to calculate and print the 20-day moving average for the last day
24 calculate_and_print_moving_average(closing_prices)
25

```

Εικόνα 7: Guide through στο extension

Τέλος, για να κάνουμε την πρόβλεψή μας, η λογική μας είναι πως εάν η τιμή βρίσκεται πάνω από το moving average αγοράζουμε, ενώ διαφορετικά πουλάμε. Επιχειρώντας να γράψουμε τον αντίστοιχο κώδικα χειροκίνητα, κάνουμε δύο συντακτικά λάθη και προφανώς εμφανίζεται σφάλμα.

```
if moving_average < closing_prices[-1]
```

```
    print("Buy")
```

```
else
```

```
    print("Sell")
```

Προκειμένου να επιλύσουμε το πρόβλημα, θα δοκιμάσουμε να δούμε τι θα γίνει εάν χρησιμοποιήσουμε και πάλι τη λειτουργία του debugging για να διορθώσουμε τα σφάλματα τα οποία κάναμε. Επιλέγουμε το προβληματικό κομμάτι του κώδικα, πιέζουμε την κατάλληλη συντόμευση ctrl+alt+x και γρήγορα λαμβάνουμε τον διορθωμένο κώδικα μαζί με την επεξήγηση των διορθώσεων.

#Changes made:

#1. Added a column at the end of the `else` statement to indicate the start of the else block.

#2. No changes made to the print statements as they are correct.

```
if moving_average < closing_prices[-1]:
```

4 Περιήγηση στις υλοποιήσεις

```
print("Buy")
```

else:

```
print("Sell")
```

Βλέπουμε λοιπόν πως μπορούμε να αξιοποιήσουμε το extension σε ένα πραγματικό σενάριο για τη δημιουργία ενός ολοκληρωμένου προγράμματος με νόημα. Έγινε σαφές πως οι τρεις λειτουργίες του μας φάνηκαν χρήσιμες καθ' όλη τη διαδικασία του development καθώς βοήθησαν στο να κατανοήσουμε, να εμπλουτίσουμε και να διορθώσουμε το πρόγραμμα εύκολα και γρήγορα.

5 Αξιολόγηση των υλοποιήσεων

5.1 Αξιολόγηση της ιστοσελίδας

Προκειμένου να συλλέξουμε τις εντυπώσεις των πραγματικών χρηστών για την ιστοσελίδα διενεργήθηκε διαδικασία αξιολόγησης στην οποία συμμετείχαν 13 άτομα. Το δείγμα των χρηστών αποτελείται από άτομα διαφόρων ηλικιών καθώς και με διαφορετικές σπουδές και διαφορετικά αντικείμενα ενασχόλησης. Η αξιολόγηση βασίστηκε στο πρότυπο System Usability Scale όπου ο κάθε χρήστης απάντησε σε 10 συγκεκριμένες προτάσεις, επιλέγοντας το κατά πόσο συμφωνεί η διαφωνεί με αυτές. Σκοπός της αξιολόγησης είναι να διευκρινίσουμε το κατά πόσο οι χρήστες έμειναν ικανοποιημένοι από την ιστοσελίδα καθώς και ποια είναι τα δυνατά και τα αδύναμα σημεία της και ποιες προοπτικές υπάρχουν για περαιτέρω βελτίωση. Οι απαντήσεις των χρηστών συνοψίζονται στα παρακάτω διαγράμματα.

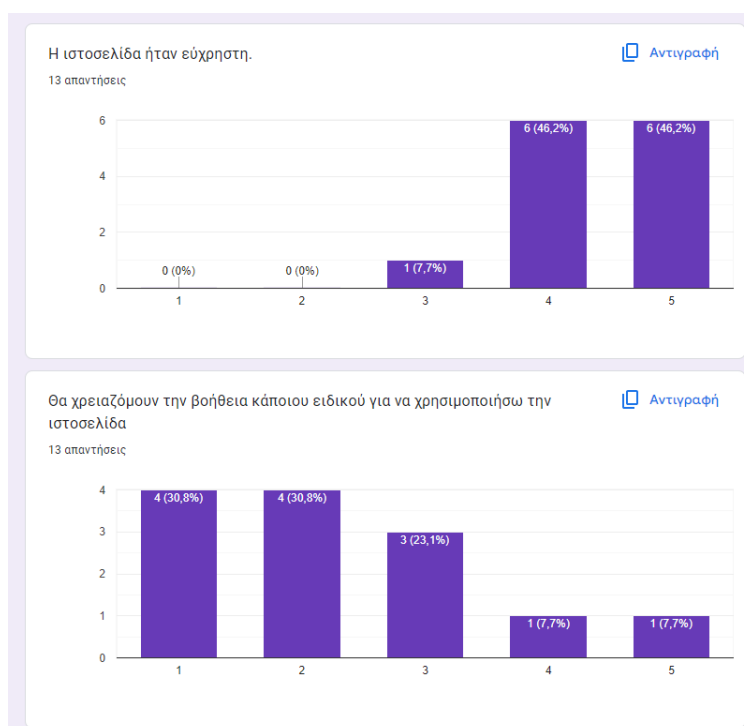


Εικόνα 8: Διαγράμματα αξιολόγησης

5 Αξιολόγηση των υλοποιήσεων

Στο πρώτο διάγραμμα οι χρήστες απαντούν στην πρόταση "Θα ήθελα να χρησιμοποιώ συχνά την ιστοσελίδα" και βλέπουμε πως στην συντριπτική τους πλειοψηφία πράγματι συμφωνούν με αυτή την πρόταση. Έτσι υποδηλώνεται πως η ιστοσελίδα αποτελεί μία ελκυστική επιλογή με σημαντική πρακτική χρησιμότητα εφόσον είναι ένα εργαλείο το οποίο οι χρήστες που ρωτήθηκαν θα ήθελαν να χρησιμοποιούν συχνά.

Στην συνέχεια οι χρήστες καλούνται να απαντήσουν στην πρόταση "Η σελίδα ήταν αχρείαστα περίπλοκη", όπου και πάλι βλέπουμε πως οι απαντήσεις των χρηστών είναι θετικές καθώς ελάχιστοι αποκρίθηκαν θετικά σε αυτήν την πρόταση δείχνοντας ότι η πλειοψηφία συμφωνεί πως η σελίδα δεν είναι περίπλοκη. Αντιθέτως είναι τόσο απλή και ξεκάθαρη όσο χρειάζεται ώστε να εξυπηρετεί τους σκοπούς της.

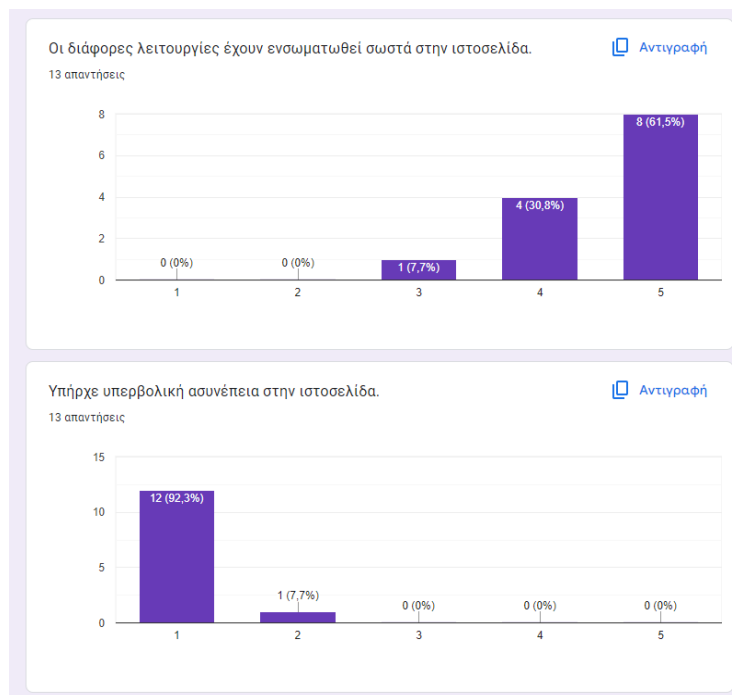


Εικόνα 9: Διαγράμματα αξιολόγησης

Η επόμενη πρόταση είναι η εξής "Η ιστοσελίδα ήταν εύχρηστη", στην οποία και πάλι η πλειοψηφία των χρηστών απάντησε πως όντως η ιστοσελίδα είναι εύχρηστη. Το αποτέλεσμα αυτό είναι λογικό και αναμενόμενο εφόσον η ευκολία χρήσης λήφθηκε σοβαρά υπόψη κατά την υλοποίηση του εργαλείου. Η ιστοσελίδα είναι έτσι δομημένη

ώστε η λειτουργικότητα η οποία παρέχει να είναι εμφανής και κατανοητή, ενώ ταυτόχρονα να μπορεί να αξιοποιηθεί με όσο το δυνατόν λιγότερα βήματα.

Ακολουθεί η πρόταση "Θα χρειαζόμουν τη βοήθεια κάποιου ειδικού για να χρησιμοποιήσω την ιστοσελίδα". Εδώ βλέπουμε πως κάποιοι χρήστες φαίνεται να είχαν κάποιες απορίες σχετικά με τη χρήση του εργαλείου. Αυτό πιθανότατα οφείλεται στην έλλειψη εξοικειώσής τους με μοντέλα τεχνητής νοημοσύνης. Κάποιες λειτουργίες του εργαλείου παρότι δεν απαιτούν γνώσεις για τη χρήση τους, ίσως να ξενίσουν σε κάποιον ο οποίος τις αντικρίζει για πρώτη φορά. Για τη βοήθεια αυτών των χρηστών θα είχε νόημα η δημιουργία μιας σελίδας βοήθειας όπου θα περιγράφονταν αναλυτικά και με πραγματικά παραδείγματα οι διάφορες λειτουργίες και ο τρόπος χρήσης τους. Ακόμα, για τον ίδιο σκοπό θα μπορούσε να συμπεριληφθεί στην ιστοσελίδα η δυνατότητα να εμφανίζονται σύντομες οδηγίες χρήσης όταν ο χρήστης κάνει hover με τον κέρσορα πάνω από ένα αντικείμενο.



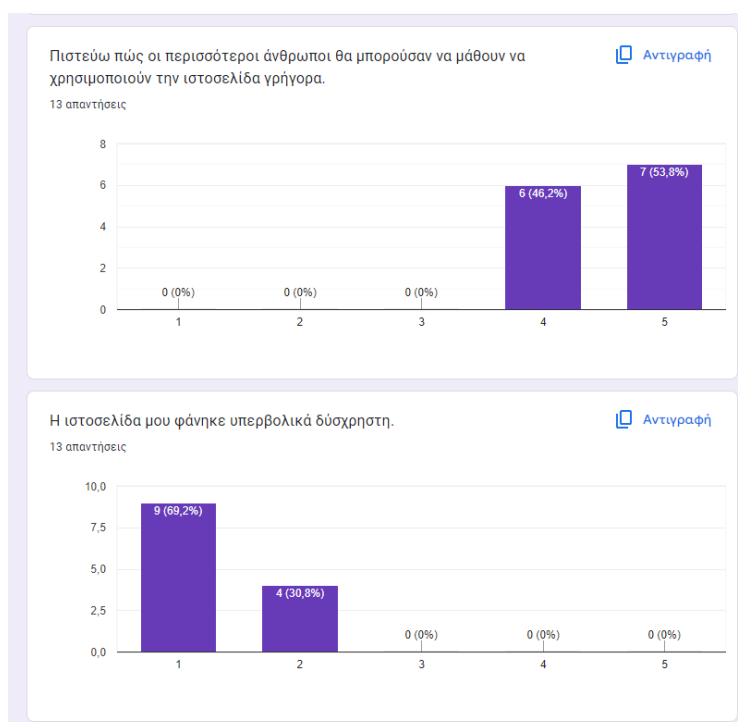
Εικόνα 10: Διαγράμματα αξιολόγησης

Έπεται η πρόταση "Οι διάφορες λειτουργίες έχουν ενσωματωθεί σωστά στην ιστοσελίδα". Για άλλη μία φορά βλέπουμε πως οι ερωτηθέντες απάντησαν θετικά

5 Αξιολόγηση των υλοποιήσεων

υποδεικνύοντας ότι το εργαλείο λειτουργεί ομαλά, δίχως προβλήματα ή σφάλματα και οι διάφορες λειτουργίες είναι εύκολο να αξιοποιηθούν και να εκτελεστούν με τον αναμενόμενο τρόπο δίχως κολλήματα.

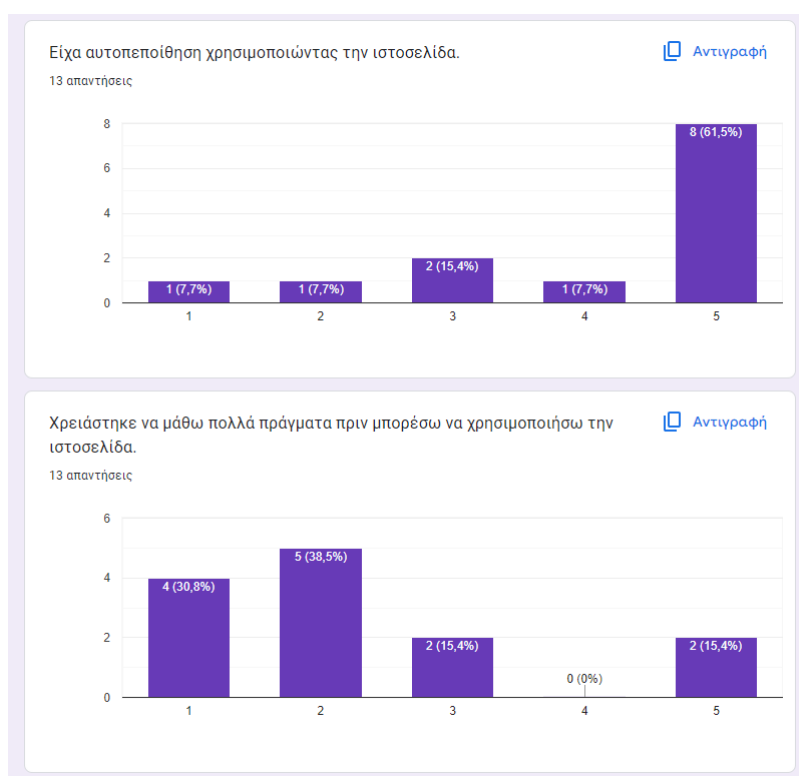
Στην ακόλουθη πρόταση "Υπήρχε υπερβολική ασυνέπεια στην ιστοσελίδα", όπου σχεδόν όλες οι αποκρίσεις συγκλίνουν στο "διαφωνώ απόλυτα". Αμέσως συμπεραίνουμε λοιπόν πως το εργαλείο διατηρεί σταθερό τον χαρακτήρα, την απόδοση και την αξιοπιστία του σε κάθε περίπτωση και σε διαφορετικά σενάρια χρήσης.



Εικόνα 11: Διαγράμματα αξιολόγησης

Η επόμενη πρόταση είναι η εξής "Πιστεύω πως οι περισσότεροι άνθρωποι θα μπορούσαν να μάθουν να χρησιμοποιούν την ιστοσελίδα γρήγορα". Όλοι οι χρήστες οι οποίοι συμμετείχαν στην αξιολόγηση συμφωνούν πως η παραπάνω πρόταση είναι αληθής. Αυτή είναι άλλη μία αναμενόμενη απάντηση καθότι το σύστημα είναι έτσι οργανωμένο και σχεδιασμένο ώστε να μπορεί να αξιοποιηθεί από οποιονδήποτε χρήστη ασχέτως του μορφωτικού του επιπέδου, δίχως να χρειάζονται πολλές εξηγήσεις. Όλες οι λειτουργίες του συστήματος είναι εύκολο να εντοπιστούν και να εκτελεστούν σε λίγα βήματα έτσι ώστε ακόμα και κάποιος ο οποίος αντικρίζει για πρώτη φορά ένα τέτοιο σύστημα να είναι σε θέση να εξοικειωθεί γρήγορα.

Στην συνέχεια οι ερωτηθέντες κλήθηκαν να αποκριθούν στην πρόταση "Η ιστοσελίδα μου φάνηκε υπερβολικά δύσχρηστη". Όπως είναι αναμενόμενο κρίνοντας και από τις προηγούμενες απαντήσεις η συντριπτική πλειοψηφία των ερωτηθέντων αποκρίθηκε πως διαφωνεί απόλυτα με αυτήν την πρόταση καταδεικνύοντας με τον πιο εμφατικό τρόπο την ευχρηστία και την απλότητα χρήσης του εργαλείου.



Εικόνα 12: Διαγράμματα αξιολόγησης

Ακολουθεί η πρόταση "Είχα αυτοπεποίθηση χρησιμοποιώντας την ιστοσελίδα". Σε αυτήν την περίπτωση παρότι η πλειοψηφία των ατόμων τα οποία συμμετείχαν στην αξιολόγηση απάντησε πως συμφωνεί απόλυτα και όντως χρησιμοποίησαν την ιστοσελίδα με αυτοπεποίθηση υπήρχαν και κάποιοι οι οποίοι διαφωνούν. Έτσι μπορούμε να συμπεράνουμε πως ίσως υπάρχουν στοιχεία στο εργαλείο τα οποία λόγω της πρωτοποριακής λειτουργικότητάς τους παραξένεψαν τους άπειρους χρήστες βάζοντάς τους σε σκέψεις. Για να εξυπηρετηθούν με περισσότερη άνεση και αυτά τα άτομα θα είχε νόημα η δημιουργία ενός βίντεο tutorial στο οποίο θα υλοποιείται ένα ή και περισσότερα πραγματικά σενάρια χρήσης του εργαλείου επιδεικνύοντας πως λειτουργεί

5 Αξιολόγηση των υλοποιήσεων

το σύστημα, έτσι ώστε ακόμα και οι πιο αρχάριοι χρήστες να αποκτήσουν μία εισαγωγική αυτοπεποίθηση και σιγουριά.

Τέλος, στην πρόταση "Χρειάστηκε να μάθω πολλά πράγματα πριν μπορέσω να χρησιμοποιήσω την ιστοσελίδα" βλέπουμε και πάλι ανάμεικτα συναισθήματα. Παρότι η πλειοψηφία των χρηστών διαφωνεί με την παραπάνω πρόταση δείχνοντας πως οποιοσδήποτε καταρτισμένος χρήστης μπορεί να την αξιοποιήσει με άνεση, υπάρχουν και κάποιοι οι οποίοι θεωρούν πως πράγματι θα χρειαζόταν να μάθουν αρκετά πράγματα πριν τη χρησιμοποιήσουν. Στην πραγματικότητα, σε πολλές περιπτώσεις, αυτό είναι μία ψευδαίσθηση η οποία δημιουργείται σε αρχάριους οι οποίοι δεν έχουν εμπειρία με τέτοια συστήματα καθότι είναι δύσκολο να αντιληφθεί και να συνειδητοποιήσει κάποιος την πρωτοποριακή τεχνολογία στην οποία βασίζονται. Εδώ θα βοηθούσε αν ο χρήστης διέθετε εξ αρχής κάποιες στοιχειώδεις γνώσεις περί των μεγάλων γλωσσικών μοντέλων και τον τρόπο που μπορούν να αξιοποιηθούν και να ενσωματωθούν στη διαδικασία συγγραφής κώδικα. Από εκεί και πέρα όσα αναφέρθηκαν παραπάνω περί tutorial και καθοδηγήσεων στο σύστημα θα μπορούσαν επίσης να βοηθήσουν.

Προκειμένου να υπολογίσουμε την επίδοση του συστήματος, σύμφωνα πάντα με το πρότυπο System Usability Scale, θα επιδώσουμε βαθμούς από το 0 έως το 4 σε κάθε μία από τις απαντήσεις των χρηστών. Έπειτα, θα εξάγουμε τον μέσο όρο της βαθμολογίας για κάθε μία από τις 10 προτάσεις, θα αθροίσουμε τους αριθμούς και θα πολλαπλασιάσουμε το σύνολο με 2,5 ώστε να καταλήξουμε στην τελική, συνολική βαθμολογία. Στην προκειμένη περίπτωση η βαθμολογία διαμορφώνεται στο 85 / 100. Σύμφωνα με το πρότυπο, ο μέσος όρος της βαθμολογίας των συστημάτων είναι 68 / 100. Αυτό σημαίνει πως η υλοποίηση αυτής της εργασίας καταφέρνει να ξεπεράσει και μάλιστα με σημαντική διαφορά τον μέσο όρο αφήνοντας ιδιαίτερα καλές εντυπώσεις στη συντριπτική πλειοψηφία των συμμετεχόντων στην έρευνα σε σημείο μάλιστα όπου πολλοί ίσως να ήταν και πρόθυμοι να το συστήσουν σε γνωστούς τους.

Συνολικά βλέπουμε πως οι χρήστες είναι αρκετά ικανοποιημένοι και ευχαριστημένοι με την ιστοσελίδα και τις υπηρεσίες που προσφέρει. Στη συντριπτική τους πλειοψηφία, τα άτομα τα οποία συμμετείχαν στην αξιολόγηση δήλωσαν ευχαριστημένα με τη λειτουργικότητα την οποία παρέχει το σύστημα και εξέφρασαν το

ενδιαφέρον τους για την περαιτέρω αξιοποίησή του. Οι κύριοι, αν και περιορισμένοι, προβληματισμοί οι οποίοι παρατηρήθηκαν σχετίζονται με την έλλειψη εξοικείωσης ορισμένων ατόμων με συστήματα και τεχνολογίες αυτού του τύπου όπως τα μεγάλα γλωσσικά μοντέλα και την ενσωμάτωση τους στη διαδικασία δημιουργίας προγραμμάτων κώδικα. Όπως αναφέρθηκε και προηγουμένως, αυτοί οι προβληματισμοί θα μπορούσαν εύκολα να καμφθούν με τη χρήση κάποιων tutorials, την παροχή κάποιων βασικών οδηγιών και την ενημέρωση των χρηστών σχετικά με τη χρήση των μεγάλων γλωσσικών μοντέλων και τον τρόπο με τον οποίο έχουν υλοποιηθεί οι διάφορες λειτουργίες του συστήματος.

6 Επίλογος

6.1 Σύνοψη και συμπεράσματα

Συνολικά, η εργασία αυτή εξετάζει το αντικείμενο του prompt engineering και τις διάφορες προσεγγίσεις που υπάρχουν, τις ενδεδειγμένες πρακτικές και τα πεδία εφαρμογής τους και εν τέλει συνεισφέρει πρακτικά στην εφαρμογή αυτών των μεθόδων μέσω της ιστοσελίδας και του extension που δημιουργήθηκε. Αρχικά εξετάστηκαν και αναδείχθηκαν τεχνικές όπως το chain of thought, το few shot αλλά και καλές πρακτικές και συστάσεις για τη χρήση μεγάλων γλωσσικών μοντέλων στην επίλυση προβλημάτων με έμφαση στη συγγραφή κώδικα. Ειδικότερα εφόσον η εργασία στοχεύει κυρίως στη χρήση αυτών των μεθόδων για τη συγγραφή κώδικα, εξετάστηκαν οι αντίστοιχες τεχνικές καθώς και τεχνικές αποσφαλμάτωσης. Επίσης εξετάστηκαν, περιγράφηκαν και αξιολογήθηκαν εργαλεία prompt engineering και prompt generation τα οποία δύνανται να βοηθήσουν προς αυτή την κατεύθυνση. Σε αυτή την κατεύθυνση δημιουργήθηκαν η ιστοσελίδα η οποία βοηθάει τους χρήστες να δημιουργήσουν εύκολα μεγάλο όγκο κώδικα χρησιμοποιώντας το ChatGPT και το extension το οποίο συνδράμει σημαντικά στη συγγραφή κώδικα μέσα από το περιβάλλον Visual Studio Code αλλά και στην αποσφαλμάτωσή του.

Συμπερασματικά, μέσα από την εργασία αναδεικνύεται η χρησιμότητα αυτών των μεθόδων, η αποτελεσματικότητα των καλών πρακτικών και των συστάσεων αλλά και ανακαλύψαμε συνολικά συμπεράσματα και κατευθύνσεις τις οποίες είναι καλό να ακολουθούμε κατά την εκπόνηση οποιασδήποτε εργασίας και ειδικότερα της συγγραφής κώδικα με χρήση τεχνητής νοημοσύνης. Πλέον διαφαίνεται πως οι μέθοδοι chain of thought και few shot δεν είναι απλώς τυπικές συστάσεις αλλά επηρεάζουν καταλυτικά την ποιότητα του παραγόμενου αποτελέσματος. Το αποτέλεσμα μπορεί να είναι πολύ πιο κοντά σε αυτό το οποίο φιλοδοξεί ο χρήστης όταν ακολουθεί τις καλές πρακτικές και τις συστάσεις όπως το να περιγράφει αναλυτικά το αντικείμενο της εργασίας και το ζητούμενο, το να παρέχει εύστοχες πληροφορίες κατευθύνοντας το μοντέλο χωρίς ωστόσο να φλυαρεί, να παρέχει αρκούντως συμπεριληπτικά παραδείγματα κ.ά.

Επίσης πολύ χρήσιμο είναι το CLEAR framework το οποίο μας συμβουλεύει να δομούμε την είσοδο στο μοντέλο με τρόπο σύντομο, λογικό, ξεκάθαρο, περιγραφικό και σε μορφή η οποία εξασφαλίζει την αμφίδρομη επικοινωνία με το μοντέλο. Είναι ολοφάνερο πλέον πως το σημαντικότερο είναι να υπάρχει συναντίληψη μεταξύ μοντέλου και χρήστη. Όταν ο χρήστης καταφέρνει να επικοινωνήσει στο μοντέλο την ουσία της σκέψης του, τότε και το μοντέλο είναι σε θέση να ανταποκριθεί με τον επιθυμητό τρόπο. Επιπλέον έχουμε εξετάσει συχνά σφάλματα στα οποία τείνουν να υποπέφτουν οι αρχάριοι χρήστες. Πολλά από αυτά σχετίζονται με την επικοινωνία με το μοντέλο η οποία τείνει να είναι είτε πρόχειρη, αντιμετωπίζοντας το ως άνθρωπο είτε να διακόπτεται πρόωρα πρώτου εξεταστούν διαφορετικές προοπτικές και εναλλακτικές ενώ ακόμα, πολλοί νέοι χρήστες δεν αντιλαμβάνονται πλήρως τις δυνατότητες και τους περιορισμούς αυτών των εργαλείων.

Σχετικά με την συγγραφή κώδικα συμπεραίνουμε πως τα ποιοτικότερα αποτελέσματα παράγονται όταν η είσοδος στο μοντέλο έχει την πιο ορθά δομημένη μορφή και ιδιαίτερα όταν η μορφή της εισόδου προσιδιάζει την μορφή ψευδοκώδικα, η απόδοση του μοντέλου βελτιώνεται αισθητά. Όπως και στις περισσότερες εργασίες έτσι και εδώ τεχνικές όπως το few shot και το chain of thought παραμένουν καταλυτικής σημασίας, ενώ για άλλη μία φορά η χρήση φυσικής γλώσσας με σκοπό να επικοινωνήσουμε στο μοντέλο με τρόπο σαφή και εύστοχο την επιθυμία μας αποτελεί τον βασικό πυλώνα αυτού του είδους prompt engineering. Γενικότερα διαφαίνεται πως με τη χρήση αυτών των τεχνικών η συγγραφή κώδικα για μία ευρεία ποικιλία χρηστών από αρχάριους έως και προχωρημένους επιταχύνεται και διευκολύνεται αισθητά. Επίσης εξερευνήθηκε η δυνατότητα αποσφαλμάτωσης χρήσει Large Language Models όπου ακόμα και εδώ οι ενδεικνυόμενες τεχνικές και κατευθύνσεις είναι εμφανές ότι διευκολύνουν τη διαδικασία.

Εν συνεχεία μελετήσαμε πολλά εργαλεία prompt engineering και prompt generation τα οποία αποσκοπούν στο να συνδράμουν τον χρήστη στο να διεκπεραιώσει τις εργασίες του βάσει των παραπάνω μεθοδολογιών πιο γρήγορα, πιο εύκολα και αποφεύγοντας τα λάθη. Ακόμα, αρκετά από αυτά στοχεύουν στο να διευρύνουν τις δυνατότητες του χρήστη και να τον καταστήσουν ικανό να επιτελέσει πιο περίπλοκες

6 Επίλογος

διεργασίες χτίζοντας πάνω στο υπάρχον μοντέλο. Πολλά από αυτά τα εργαλεία είναι απλοϊκά χωρίς να προσφέρουν στον χρήστη καμία αναβαθμισμένη εμπειρία. Άλλα πιο ενδιαφέροντα εργαλεία πραγματοποιούν αναδιατύπωση της εισόδου σε πιο περιγραφική μορφή ή και customization της εισόδου.

Άλλα εργαλεία μας επιτρέπουν να εισάγουμε μεταβλητές και σύνολα δεδομένων μέσα στο prompt, να εξετάσουμε διαφορετικές εκδοχές του prompt στην επίλυση ενός προβλήματος, να εκτελέσουμε ένα πρόγραμμα επαναληπτικά πάνω σε ένα μεγάλο όγκο δεδομένων, να δημιουργήσουμε AI chains με αλληπάλλληλες κλήσεις σε νευρωνικά δίκτυα κ.ά. Ακόμα πολύ ενδιαφέρουσα είναι η δυνατότητα που μας δίνουν αυτά τα εργαλεία να δημιουργήσουμε ολόκληρες εφαρμογές και πράκτορες βασισμένους σε νευρωνικά δίκτυα. Συνολικά τα εργαλεία αυτά μπορούν να εφαρμοστούν σε πληθώρα διαφορετικών περιπτώσεων από αρχάριους έως και απαιτητικούς χρήστες καθώς και να αξιοποιηθούν σε πολλά projects αυξάνοντας κατακόρυφα την παραγωγικότητα και την ευελιξία.

Σε συνέχεια των προαναφερθέντων, έρχονται να προσφέρουν και οι υλοποιήσεις οι οποίες πραγματοποιήθηκαν στα πλαίσια αυτής της εργασίας. Η ιστοσελίδα καταφέρνει να βοηθήσει αρχάριους αλλά και προχωρημένους χρήστες στο να αξιοποιήσουν με τον βέλτιστο δυνατό τρόπο τα μεγάλα γλωσσικά μοντέλα είτε αυτό αφορά μία απλή επικοινωνία είτε την ολοκλήρωση μιας απαιτητικής και πολύπλοκης παραγωγικής διαδικασίας με πολλά δεδομένα. Ο μέσος αρχάριος χρήστης θα μπορέσει μέσω της ιστοσελίδας να οργανώσει και να διευρύνει τη σκέψη του και να την επικοινωνήσει αποτελεσματικά στο μοντέλο. Από την άλλη ένας πιο έμπειρος και απαιτητικός χρήστης μπορεί να μεταφέρει με ακρίβεια τα χαρακτηριστικά της εργασίας του στο μοντέλο, να τα συζητήσει μαζί του, να αναλύσουν ακόμα και τις πιο περίπλοκες λεπτομέρειες και εν τέλει να του ζητήσει να την υλοποιήσει ακριβώς με τον τρόπο που την οραματίζεται.

Παράλληλα, μέσω της ιστοσελίδας οποιοσδήποτε χρήστης μπορεί να επικοινωνήσει με εναλλακτικές οδούς με το μοντέλο όπως η παροχή εικόνων και αρχείων. Αυτές οι λειτουργίες όχι μόνο διευκολύνουν και επιταχύνουν τη διεκπεραίωση της εργασίας αλλά χτίζουν πάνω στις υπάρχουσες δυνατότητες των μοντέλων ώστε να τα αξιοποιήσουν στον μέγιστο βαθμό για να παρέχουν στον χρήστη μία πιο ολοκληρωμένη

και αναπτυγμένη εμπειρία η οποία θα τον φέρει πιο κοντά στον στόχο του και μάλιστα μπορεί ακόμα και να του διευρύνει τους ορίζοντες.

Όσον αφορά το extension, πηγαίνει ένα βήμα παραπέρα την αξιοποίηση τέτοιου είδους μοντέλων καθώς ενσωματώνεται μέσα σε ένα γνωστό προγραμματιστικό περιβάλλον. Εκεί το extension είναι έτοιμο να συνδράμει τον χρήστη στη συγγραφή προγραμμάτων, στη διόρθωση καθώς και στην ανάλυσή τους αποτελώντας εξαιρετικό και χαρακτηριστικό παράδειγμα της αξιοποίησης των μεγάλων γλωσσικών μοντέλων σε ολοκληρωμένες εφαρμογές και με αυτόν τον τρόπο της διεύρυνσης των δυνατοτήτων τους και της ανάπτυξης εφαρμογών στηριζόμενων σε αυτά.

Συνολικά η εργασία καταφέρνει με αυτόν τον τρόπο όχι μόνο να μελετήσει και να αναλύσει τις καλές πρακτικές μεθόδους για τη χρήση των μεγάλων γλωσσικών μοντέλων με έμφαση στην συγγραφή κώδικα αλλά με τις υλοποιήσεις καταφέρνουμε και να εφαρμόσουμε όσα μάθαμε σε ένα ολοκληρωμένο περιβάλλον αυξάνοντας κατακόρυφα την αποτελεσματικότητα αυτών των εργαλείων.

6.2 Όρια και περιορισμοί της έρευνας

Θα πρέπει να σημειωθεί πως η έρευνα εξειδικεύεται στη χρήση μεγάλων γλωσσικών μοντέλων για την συγγραφή κώδικα και παρότι πολλά από τα συμπεράσματα και τις κατευθύνσεις που παρέχονται μπορούμε να τα εφαρμόσουμε και σε οποιαδήποτε άλλη εργασία με τη χρήση τέτοιων μοντέλων, οι υλοποιήσεις της εργασίας δηλαδή η ιστοσελίδα και το extension είναι εξειδικευμένα στην συγγραφή κώδικα και πιθανότατα δεν θα έχουν την ανάλογη απόδοση εάν προσπαθήσουμε να τα αξιοποιήσουμε για άλλου τύπου εργασίες. Παράλληλα στην εργασία επιδιώκεται η όσο το δυνατόν συμπεριληπτικότητα όλων των ενδεχόμενων χρηστών τέτοιων μοντέλων καθώς και των υλοποιήσεων εργασίας σε οποιοδήποτε πιθανό σενάριο. Αυτό σημαίνει πως για την περαιτέρω αξιολόγηση και τροποποίηση της εργασίας θα ήταν πολύ δύσκολο λόγω του ευρέως κοινού στο οποίο απευθύνεται, η συλλογή και αξιοποίηση προσωποποιημένων δεδομένων και χαρακτηριστικών. Ακόμα είναι σημαντικό να λάβουμε υπόψη μας πως η

6 Επίλογος

τεχνητή νοημοσύνη εξελίσσεται και αναπτύσσεται με αλματώδη ταχύτητα κάτι το οποίο σημαίνει πως οτιδήποτε έχει γραφτεί σε αυτή την εργασία μέσα σε μερικούς μήνες μπορεί ήδη να θεωρηθεί, τουλάχιστον μερικώς, παρωχημένο. Παρόλα αυτά έχει γίνει προσπάθεια για την εξαγωγή μακροπρόθεσμων κατευθύνσεων και για την παροχή συμπερασμάτων τα οποία θα φανούν χρήσιμα σε όλους τους χρήστες ανεξαρτήτως του πλαισίου χρήσης του μοντέλου.

6.3 Μελλοντικές Επεκτάσεις

Οι δυνητικές μελλοντικές προεκτάσεις της έρευνας είναι πάρα πολλές και πάρα πολύ ενδιαφέρουσες. Για παράδειγμα μία ιδέα για περαιτέρω ανάπτυξη των εφαρμογών θα μπορούσε να είναι η συλλογή ιστορικού του χρήστη και η απάντηση σε μελλοντικά ερωτήματα με βάση αυτό το ιστορικό. Ακόμα πιο ενδιαφέρουσα θα ήταν η διενέργεια πειραμάτων και γκάλοπ βάσει των υλοποιήσεων της εργασίας με σκοπό τη συλλογή εντυπώσεων από τους χρήστες και την περαιτέρω βελτίωση των εφαρμογών με βάση αυτές τις κατευθύνσεις. Πέρα από αυτά, οι εφαρμογές των υλοποιήσεων θα μπορούσαν να αναπτυχθούν με τη δημιουργία αντίστοιχων extensions για περισσότερα integrated development environments και την μεταφορά της ιστοσελίδας σε κάποια εφαρμογή για κινητά τηλέφωνα.

Επιπρόσθετα, στα πλαίσια της ιστοσελίδας θα ήταν πολύ ενδιαφέρον να αναπτυχθεί μία κοινότητα και κάποιο marketplace όπου ο κάθε χρήστης θα μπορεί να δημιουργεί τον δικό του fine tuned assistant με τις δικές του εντολές και τον δικό του τρόπο απόκρισης και να διαθέτει αυτόν τον assistant για χρήση στο marketplace. Όσον αφορά το τμήμα του fine tuning στο οποίο βασίζονται πολλά από τα μοντέλα τα οποία αξιοποιούνται στις υλοποιήσεις, θα ήταν πολύ χρήσιμο έπειτα από ένα διάστημα χρήσης της ιστοσελίδας και του extension από αρκετούς χρήστες να συλλεχθούν τα prompts των χρηστών, να συμπληρωθούν οι επιθυμητές απαντήσεις από τους developers και έπειτα αυτές οι είσοδοι και οι αντίστοιχες έξοδοι να παρασχεθούν στο μοντέλο προκειμένου να γίνει επιπλέον fine tuning μεγάλης κλίμακας με πολλά πραγματικά δεδομένα από πραγματικούς χρήστες.

Επίσης, μία ακόμα πολύ ενδιαφέρουσα και πρακτική προσθήκη θα μπορούσε κάλλιστα να ενσωματωθεί στην ιστοσελίδα. Αυτή αφορά τη δημιουργία ενός toolbar στο οποίο ο χρήστης θα μπορεί να αποθηκεύει έτοιμα prompts τα οποία του άρεσαν, τα θεωρεί χρήσιμα και σκοπεύει να τα επαναχρησιμοποιήσει στο μέλλον. Με αυτόν τον τρόπο θα μπορεί ανά πάσα στιγμή, πιέζοντας απλώς το αντίστοιχο πλήκτρο της εργαλειοθήκης, να θέτει το έτοιμο prompt ως είσοδο στο μοντέλο, αντιμετωπίζοντάς το πλέον ως υπηρεσία. Ακόμα, το prompt αυτό θα μπορούσε να δέχεται και μεταβλητές ώστε να λειτουργεί και ως συνάρτηση έτοιμη για αξιοποίηση σε διαφορετικά προβλήματα.

Πέρα από αυτά, στο θεωρητικό κομμάτι προκειμένου να αντιληφθούμε ποιο είναι το ύφος prompt στο οποίο τα μοντέλα απαντούν καλύτερα, θα ήταν πραγματικά χρήσιμο να ζητήσουμε από ένα τέτοιο μοντέλο να δημιουργήσει πολλά prompts το καθένα με διαφορετικό ύφος, να τα τροφοδοτήσουμε όλα στο εργαλείο και να αξιολογήσουμε τις απαντήσεις του. Για να γίνει αυτό θα μπορούσαμε είτε να καθοδηγήσουμε το μοντέλο να παράγει prompts με βάση τις γνωστές τεχνικές και συστάσεις είτε να σκεφτεί δημιουργικά συντάσσοντας δικά του prompts οποία ακολουθούν ανεξαιρεύνητα πρότυπα.

Σε κάθε περίπτωση, αυτές είναι μόνο λίγες από τις δυνατότητες εξέλιξης και ανάπτυξης που έχει και το θεωρητικό αλλά και το τεχνικό κομμάτι στο συγκεκριμένο αντικείμενο. Η τεχνητή νοημοσύνη προοδεύει με εντυπωσιακό ρυθμό και οι μελλοντικές προεκτάσεις φαντάζουν ανεξάντλητες ακόμα και σε ορίζοντα λίγων μηνών.

Βιβλιογραφία - Αρθρογραφία

- [1] S. Wang και P. Jin, *A Brief Summary of Prompting in Using Gpt Models*. 2023. doi: 10.32388/IMZI2Q.
- [2] M. Chen κ.ά., ‘Evaluating Large Language Models Trained on Code’. arXiv, 14 Ιούλιος 2021. doi: 10.48550/arXiv.2107.03374.
- [3] R. Lou, K. Zhang, και W. Yin, ‘Is Prompt All You Need? No. A Comprehensive and Broader View of Instruction Learning’. arXiv, 2 Μάιος 2023. doi: 10.48550/arXiv.2303.10475.
- [4] C. Ge κ.ά., ‘Domain Adaptation via Prompt Learning’. arXiv, 14 Φεβρουάριος 2022. doi: 10.48550/arXiv.2202.06687.
- [5] S. Diao κ.ά., ‘Black-box Prompt Learning for Pre-trained Language Models’. arXiv, 23 Φεβρουάριος 2023. doi: 10.48550/arXiv.2201.08531.
- [6] Y. Zhu, X. Zhou, J. Qiang, Y. Li, Y. Yuan, και X. Wu, ‘Prompt-Learning for Short Text Classification’. arXiv, 31 Μάρτιος 2022. doi: 10.48550/arXiv.2202.11345.
- [7] A. Madotto, Z. Lin, G. I. Winata, και P. Fung, ‘Few-Shot Bot: Prompt-Based Learning for Dialogue Systems’. arXiv, 15 Οκτώβριος 2021. doi: 10.48550/arXiv.2110.08118.
- [8] N. Ding κ.ά., ‘OpenPrompt: An Open-source Framework for Prompt-learning’. arXiv, 2 Νοέμβριος 2021. doi: 10.48550/arXiv.2111.01998.
- [9] P. Lu κ.ά., ‘Dynamic Prompt Learning via Policy Gradient for Semi-structured Mathematical Reasoning’. arXiv, 2 Μάρτιος 2023. doi: 10.48550/arXiv.2209.14610.
- [10] M. Suzgun κ.ά., ‘Challenging BIG-Bench Tasks and Whether Chain-of-Thought Can Solve Them’. arXiv, 17 Οκτώβριος 2022. doi: 10.48550/arXiv.2210.09261.
- [11] J. Liu, D. Shen, Y. Zhang, B. Dolan, L. Carin, και W. Chen, ‘What Makes Good In-Context Examples for GPT-3?’, στο *Proceedings of Deep Learning Inside Out (DeeLIO 2022): The 3rd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, Dublin, Ireland and Online: Association for Computational Linguistics, Φεβρουαρίου 2022, σσ. 100–114. doi: 10.18653/v1/2022.deelio-1.10.

- [12] P. Bareiß, B. Souza, M. d'Amorim, και M. Pradel, 'Code Generation Tools (Almost) for Free? A Study of Few-Shot, Pre-Trained Language Models on Code'. arXiv, 12 Ιούνιος 2022. doi: 10.48550/arXiv.2206.01335.
- [13] L. Reynolds και K. McDonell, 'Prompt Programming for Large Language Models: Beyond the Few-Shot Paradigm'. arXiv, 15 Φεβρουάριος 2021. doi: 10.48550/arXiv.2102.07350.
- [14] T. Z. Zhao, E. Wallace, S. Feng, D. Klein, και S. Singh, 'Calibrate Before Use: Improving Few-Shot Performance of Language Models'. arXiv, 10 Ιούνιος 2021. doi: 10.48550/arXiv.2102.09690.
- [15] Q. Zheng κ.ά., 'CodeGeeX: A Pre-Trained Model for Code Generation with Multilingual Evaluations on HumanEval-X'. arXiv, 30 Μάρτιος 2023. doi: 10.48550/arXiv.2303.17568.
- [16] P. Liu, W. Yuan, J. Fu, Z. Jiang, H. Hayashi, και G. Neubig, 'Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing'. arXiv, 28 Ιούλιος 2021. doi: 10.48550/arXiv.2107.13586.
- [17] Y. Liu κ.ά., 'Jailbreaking ChatGPT via Prompt Engineering: An Empirical Study'. arXiv, 23 Μάιος 2023. doi: 10.48550/arXiv.2305.13860.
- [18] A. Svendsen και B. Garvey, 'Prompt-engineering testing ChatGPT4 and Bard for assessing Generative-AI efficacy to support decision-making'. Rochester, NY, 1 Ιούνιος 2023. doi: 10.2139/ssrn.4495320.
- [19] L. S. Lo, 'The Art and Science of Prompt Engineering: A New Literacy in the Information Age', *Internet Reference Services Quarterly*, τ. 0, τχ. 0, σσ. 1–8, Ιουνίου 2023, doi: 10.1080/10875301.2023.2227621.
- [20] L. S. Lo, 'The CLEAR path: A framework for enhancing information literacy through prompt engineering', *The Journal of Academic Librarianship*, τ. 49, τχ. 4, σ. 102720, Ιουλίου 2023, doi: 10.1016/j.acalib.2023.102720.
- [21] R. Clariso και J. Cabot, 'Model-Driven Prompt Engineering'.
- [22] A. K. Gao, 'Prompt Engineering for Large Language Models'.
- [23] J. D. Zamfirescu-Pereira, R. Y. Wong, B. Hartmann, και Q. Yang, 'Why Johnny Can't Prompt: How Non-AI Experts Try (and Fail) to Design LLM Prompts', στο *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*,

6 Επίλογος

- στο CHI '23. New York, NY, USA: Association for Computing Machinery, Δεκεμβρίου 2023, σσ. 1–21. doi: 10.1145/3544548.3581388.
- [24] L. Giray, ‘Prompt Engineering with ChatGPT: A Guide for Academic Writers’, *Ann Biomed Eng*, Ιουνίου 2023, doi: 10.1007/s10439-023-03272-4.
- [25] D. C. Schmidt, J. Spencer-Smith, Q. Fu, και J. White, ‘Cataloging Prompt Patterns to Enhance the Discipline of Prompt Engineering’.
- [26] J. White κ.ά., ‘A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT’. arXiv, 21 Φεβρουάριος 2023. doi: 10.48550/arXiv.2302.11382.
- [27] Y. Feng, S. Vanam, M. Cherukupally, W. Zheng, M. Qiu, και H. Chen, ‘Investigating Code Generation Performance of Chat-GPT with Crowdsourcing Social Data’, στο *Proceedings of the 47th IEEE Computer Software and Applications Conference, 2023*, σσ. 1–10.
- [28] S. Imai, ‘Is GitHub copilot a substitute for human pair-programming? an empirical study’, στο *Proceedings of the ACM/IEEE 44th International Conference on Software Engineering: Companion Proceedings*, στο ICSE '22. New York, NY, USA: Association for Computing Machinery, Ιουλίου 2022, σσ. 319–321. doi: 10.1145/3510454.3522684.
- [29] J. White, S. Hays, Q. Fu, J. Spencer-Smith, και D. C. Schmidt, ‘ChatGPT Prompt Patterns for Improving Code Quality, Refactoring, Requirements Elicitation, and Software Design’. arXiv, 11 Μάρτιος 2023. doi: 10.48550/arXiv.2303.07839.
- [30] P. Denny, V. Kumar, και N. Giacaman, ‘Conversing with Copilot: Exploring Prompt Engineering for Solving CS1 Problems Using Natural Language’, στο *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1*, στο SIGCSE 2023. New York, NY, USA: Association for Computing Machinery, Νοεμβρίου 2023, σσ. 1136–1142. doi: 10.1145/3545945.3569823.
- [31] C. Liu κ.ά., ‘Improving ChatGPT Prompt for Code Generation’. arXiv, 15 Μάιος 2023. doi: 10.48550/arXiv.2305.08360.
- [32] B. Yetiştirilen, I. Özsoy, M. Ayerdem, και E. Tüzün, ‘Evaluating the Code Quality of AI-Assisted Code Generation Tools: An Empirical Study on GitHub Copilot, Amazon CodeWhisperer, and ChatGPT’. arXiv, 21 Απρίλιος 2023. doi: 10.48550/arXiv.2304.10778.

- [33] Y. Li κ.ά., ‘Competition-level code generation with AlphaCode’, *Science*, τ. 378, τχ. 6624, σσ. 1092–1097, Δεκεμβρίου 2022, doi: 10.1126/science.abq1158.
- [34] I. Trummer, ‘CodexDB: Generating Code for Processing SQL Queries using GPT-3 Codex’. arXiv, 19 Απρίλιος 2022. doi: 10.48550/arXiv.2204.08941.
- [35] S. Abukhalaf, M. Hamdaqa, και F. Khomh, ‘On Codex Prompt Engineering for OCL Generation: An Empirical Study’. arXiv, 28 Μάρτιος 2023. doi: 10.48550/arXiv.2303.16244.
- [36] I. Trummer, ‘CodexDB: synthesizing code for query processing from natural language instructions using GPT-3 codex’, *Proc. VLDB Endow.*, τ. 15, τχ. 11, σσ. 2921–2928, Απριλίου 2022, doi: 10.14778/3551793.3551841.
- [37] P. Vaithilingam, T. Zhang, και E. L. Glassman, ‘Expectation vs. Experience: Evaluating the Usability of Code Generation Tools Powered by Large Language Models’, στο *Extended Abstracts of the 2022 CHI Conference on Human Factors in Computing Systems*, στο CHI EA '22. New York, NY, USA: Association for Computing Machinery, Δεκεμβρίου 2022, σσ. 1–7. doi: 10.1145/3491101.3519665.
- [38] M. L. Siddiq, A. Samee, S. R. Azgor, A. Haider, S. I. Sawraz, και J. C. S. Santos, ‘Zero-shot Prompting for Code Complexity Prediction Using GitHub Copilot’.
- [39] A. Madaan, S. Zhou, U. Alon, Y. Yang, και G. Neubig, ‘Language Models of Code are Few-Shot Commonsense Learners’. arXiv, 6 Δεκέμβριος 2022. doi: 10.48550/arXiv.2210.07128.
- [40] X. Wang, S. Li, και H. Ji, ‘Code4Struct: Code Generation for Few-Shot Event Structure Prediction’. arXiv, 24 Μάιος 2023. doi: 10.48550/arXiv.2210.12810.
- [41] A. Elnashar, M. Moundas, D. C. Schimdt, J. Spencer-Smith, και J. White, ‘Prompt Engineering of ChatGPT to Improve Generated Code & Runtime Performance Compared with the Top-Voted Human Solutions’.
- [42] M. Yasunaga κ.ά., ‘Large Language Models as Analogical Reasoners’. arXiv, 2 Οκτώβριος 2023. Ημερομηνία πρόσβασης: 9 Οκτώβριος 2023. [Έκδοση σε ψηφιακή μορφή]. Διαθέσιμο στο: <http://arxiv.org/abs/2310.01714>
- [43] S. Fakhoury, S. Chakraborty, M. Musuvathi, και S. K. Lahiri, ‘Towards Generating Functionally Correct Code Edits from Natural Language Issue Descriptions’. arXiv, 7 Απρίλιος 2023. doi: 10.48550/arXiv.2304.03816.
- [44] P. Zhao, ‘An Empirical Study on Using Codex for Automated Program Repair’.

- [45] Y. Wu, Z. Li, J. M. Zhang, M. Papadakis, M. Harman, και Y. Liu, ‘Large Language Models in Fault Localisation’. arXiv, 2 Οκτώβριος 2023. Ημερομηνία πρόσβασης: 8 Οκτώβριος 2023. [Έκδοση σε ψηφιακή μορφή]. Διαθέσιμο στο: <http://arxiv.org/abs/2308.15276>
- [46] H. Dang, L. Mecke, F. Lehmann, S. Goller, και D. Buschek, ‘How to Prompt? Opportunities and Challenges of Zero- and Few-Shot Learning for Human-AI Interaction in Creative Applications of Generative Models’. arXiv, 3 Σεπτέμβριος 2022. Ημερομηνία πρόσβασης: 19 Ιούλιος 2023. [Έκδοση σε ψηφιακή μορφή]. Διαθέσιμο στο: <http://arxiv.org/abs/2209.01390>
- [47] H. Walid, *Unlocking the Potential of ChatGPT: A Comprehensive Exploration of Its Applications, Advantages, Limitations, and Future Directions in Natural Language Processing*. 2023. doi: 10.13140/RG.2.2.23433.11360/3.
- [48] D. Shrivastava, H. Larochelle, και D. Tarlow, ‘Repository-Level Prompt Generation for Large Language Models of Code’, στο *Proceedings of the 40th International Conference on Machine Learning*, PMLR, Ιουλίου 2023, σσ. 31693–31715. Ημερομηνία πρόσβασης: 13 Ιούλιος 2023. [Έκδοση σε ψηφιακή μορφή]. Διαθέσιμο στο: <https://proceedings.mlr.press/v202/shrivastava23a.html>
- [49] X.-Y. Li, J.-T. Xue, Z. Xie, και M. Li, ‘Think Outside the Code: Brainstorming Boosts Large Language Models in Code Generation’. arXiv, 17 Μάιος 2023. doi: 10.48550/arXiv.2305.10679.
- [50] T. Shin, Y. Razeghi, R. L. Logan IV, E. Wallace, και S. Singh, ‘AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts’. arXiv, 7 Νοέμβριος 2020. doi: 10.48550/arXiv.2010.15980.
- [51] ‘Paraphrasing tool ► NeuralWriter - Free online paraphrase tool for all languages’. Ημερομηνία πρόσβασης: 14 Δεκέμβριος 2023. [Έκδοση σε ψηφιακή μορφή]. Διαθέσιμο στο: <https://neuralwriter.com/>
- [52] ‘Eye for AI - Eye for AI’. Ημερομηνία πρόσβασης: 14 Δεκέμβριος 2023. [Έκδοση σε ψηφιακή μορφή]. Διαθέσιμο στο: <https://eyeforai.xyz/>
- [53] FlowGPT, ‘FlowGPT - The Ultimate Library of ChatGPT Prompts | Discover, Share, and Discuss with a Vibrant Community’. Ημερομηνία πρόσβασης: 14 Δεκέμβριος 2023. [Έκδοση σε ψηφιακή μορφή]. Διαθέσιμο στο: <https://flowgpt.com>

- [54] Y. Zhang, H. Fei, D. Li, και P. Li, ‘PromptGen: Automatically Generate Prompts using Generative Models’, στο *Findings of the Association for Computational Linguistics: NAACL 2022*, M. Carpuat, M.-C. de Marneffe, και I. V. Meza Ruiz, Επιμ., Seattle, United States: Association for Computational Linguistics, Απριλίου 2022, σσ. 30–37. doi: 10.18653/v1/2022.findings-naacl.3.
- [55] ‘Prompt Refine’. Ημερομηνία πρόσβασης: 14 Δεκέμβριος 2023. [Έκδοση σε ψηφιακή μορφή]. Διαθέσιμο στο: <https://www.promptrefine.com/prompt/new>
- [56] ‘Prompter’. Ημερομηνία πρόσβασης: 14 Δεκέμβριος 2023. [Έκδοση σε ψηφιακή μορφή]. Διαθέσιμο στο: <https://prompter.engineer/>
- [57] ‘BetterPrompt.xyz’, BetterPrompt. Ημερομηνία πρόσβασης: 14 Δεκέμβριος 2023. [Έκδοση σε ψηφιακή μορφή]. Διαθέσιμο στο: <https://www.betterprompt.xyz/>
- [58] ‘Knit - A better AI playground for prompt designers’. Ημερομηνία πρόσβασης: 14 Δεκέμβριος 2023. [Έκδοση σε ψηφιακή μορφή]. Διαθέσιμο στο: <https://promptknit.com/>
- [59] ‘Build better Prompts with custom data’, Wale. Ημερομηνία πρόσβασης: 14 Δεκέμβριος 2023. [Έκδοση σε ψηφιακή μορφή]. Διαθέσιμο στο: <https://trywale.com/>
- [60] ‘PROMPTMETHEUS · Prompt IDE’, PROMPTMETHEUS. Ημερομηνία πρόσβασης: 14 Δεκέμβριος 2023. [Έκδοση σε ψηφιακή μορφή]. Διαθέσιμο στο: <https://promptmetheus.com/>
- [61] S. H. Bach κ.ά., ‘PromptSource: An Integrated Development Environment and Repository for Natural Language Prompts’. arXiv, 29 Μάρτιος 2022. doi: 10.48550/arXiv.2202.01279.
- [62] Y. Cheng, J. Chen, Q. Huang, Z. Xing, X. Xu, και Q. Lu, ‘Prompt Sapper: A LLM-Empowered Production Tool for Building AI Chains’. arXiv, 21 Ιούνιος 2023. doi: 10.48550/arXiv.2306.12028.
- [63] T. Wu κ.ά., ‘PromptChainer: Chaining Large Language Model Prompts through Visual Programming’, στο *Extended Abstracts of the 2022 CHI Conference on Human Factors in Computing Systems*, στο CHI EA '22. New York, NY, USA: Association for Computing Machinery, Δεκεμβρίου 2022, σσ. 1–10. doi: 10.1145/3491101.3519729.

- [64] S. Shim, P. Patil, R. R. Yadav, A. Shinde, και V. Devale, ‘DeeperCoder: Code Generation Using Machine Learning’, στο *2020 10th Annual Computing and Communication Workshop and Conference (CCWC)*, Ιανουαρίου 2020, σσ. 0194–0199. doi: 10.1109/CCWC47524.2020.9031149.
- [65] ‘outlines-dev/outlines: Guided Text Generation’. Ημερομηνία πρόσβασης: 14 Δεκέμβριος 2023. [Έκδοση σε ψηφιακή μορφή]. Διαθέσιμο στο: <https://github.com/outlines-dev/outlines>
- [66] ‘Parea AI’. Ημερομηνία πρόσβασης: 14 Δεκέμβριος 2023. [Έκδοση σε ψηφιακή μορφή]. Διαθέσιμο στο: <https://app.parea.ai/>
- [67] ‘Dust - Secure AI assistant with your company’s knowledge’. Ημερομηνία πρόσβασης: 14 Δεκέμβριος 2023. [Έκδοση σε ψηφιακή μορφή]. Διαθέσιμο στο: <https://dust.tt/>
- [68] ‘SmartGPT - Enhance ChatGPT Prompts in 1-Click’, SmartGPT. Ημερομηνία πρόσβασης: 14 Δεκέμβριος 2023. [Έκδοση σε ψηφιακή μορφή]. Διαθέσιμο στο: <https://getsmartgpt.com/>
- [69] ‘Promptist - a Hugging Face Space by microsoft’. Ημερομηνία πρόσβασης: 14 Δεκέμβριος 2023. [Έκδοση σε ψηφιακή μορφή]. Διαθέσιμο στο: <https://huggingface.co/spaces/microsoft/Promptist>
- [70] ‘PromptPerfect - Elevate Your Prompts to Perfection. Prompt Engineering, Optimizing, Debugging and Hosting.’ Ημερομηνία πρόσβασης: 14 Δεκέμβριος 2023. [Έκδοση σε ψηφιακή μορφή]. Διαθέσιμο στο: <https://promptperfect.jina.ai/>
- [71] H. Strobelt κ.ά., ‘Interactive and Visual Prompt Engineering for Ad-hoc Task Adaptation with Large Language Models’, *IEEE Transactions on Visualization and Computer Graphics*, τ. 29, τχ. 1, σσ. 1146–1156, Ιανουαρίου 2023, doi: 10.1109/TVCG.2022.3209479.
- [72] F. Kio. "STEPPING UP THE PROMPT GAME: REQUIREMENTS ENGINEERING AS A CATALYST FOR EFFECTIVE CODE GENERATION." (2023).

Ιστοσελίδες

- [51] ‘Paraphrasing tool ► NeuralWriter - Free online paraphrase tool for all languages’. Ημερομηνία πρόσβασης: 14 Δεκέμβριος 2023. [Έκδοση σε ψηφιακή μορφή]. Διαθέσιμο στο: <https://neuralwriter.com/>
- [52] ‘Eye for AI - Eye for AI’. Ημερομηνία πρόσβασης: 14 Δεκέμβριος 2023. [Έκδοση σε ψηφιακή μορφή]. Διαθέσιμο στο: <https://eyeforai.xyz/>
- [53] FlowGPT, ‘FlowGPT - The Ultimate Library of ChatGPT Prompts | Discover, Share, and Discuss with a Vibrant Community’. Ημερομηνία πρόσβασης: 14 Δεκέμβριος 2023. [Έκδοση σε ψηφιακή μορφή]. Διαθέσιμο στο: <https://flowgpt.com>
- [55] ‘Prompt Refine’. Ημερομηνία πρόσβασης: 14 Δεκέμβριος 2023. [Έκδοση σε ψηφιακή μορφή]. Διαθέσιμο στο: <https://www.promptrefine.com/prompt/new>
- [56] ‘Prompter’. Ημερομηνία πρόσβασης: 14 Δεκέμβριος 2023. [Έκδοση σε ψηφιακή μορφή]. Διαθέσιμο στο: <https://prompter.engineer/>
- [57] ‘BetterPrompt.xyz’, BetterPrompt. Ημερομηνία πρόσβασης: 14 Δεκέμβριος 2023. [Έκδοση σε ψηφιακή μορφή]. Διαθέσιμο στο: <https://www.betterprompt.xyz/>
- [58] ‘Knit - A better AI playground for prompt designers’. Ημερομηνία πρόσβασης: 14 Δεκέμβριος 2023. [Έκδοση σε ψηφιακή μορφή]. Διαθέσιμο στο: <https://promptknit.com/>
- [59] ‘Build better Prompts with custom data’, Wale. Ημερομηνία πρόσβασης: 14 Δεκέμβριος 2023. [Έκδοση σε ψηφιακή μορφή]. Διαθέσιμο στο: <https://trywale.com/>
- [60] ‘PROMPTMETHEUS · Prompt IDE’, PROMPTMETHEUS. Ημερομηνία πρόσβασης: 14 Δεκέμβριος 2023. [Έκδοση σε ψηφιακή μορφή]. Διαθέσιμο στο: <https://promptmetheus.com/>
- [65] ‘outlines-dev/outlines: Guided Text Generation’. Ημερομηνία πρόσβασης: 14 Δεκέμβριος 2023. [Έκδοση σε ψηφιακή μορφή]. Διαθέσιμο στο: <https://github.com/outlines-dev/outlines>
- [66] ‘Parea AI’. Ημερομηνία πρόσβασης: 14 Δεκέμβριος 2023. [Έκδοση σε ψηφιακή μορφή]. Διαθέσιμο στο: <https://app.parea.ai/>

6 Επίλογος

- [67] ‘Dust - Secure AI assistant with your company’s knowledge’. Ημερομηνία πρόσβασης: 14 Δεκέμβριος 2023. [Έκδοση σε ψηφιακή μορφή]. Διαθέσιμο στο: <https://dust.tt/>
- [68] ‘SmartGPT - Enhance ChatGPT Prompts in 1-Click’, SmartGPT. Ημερομηνία πρόσβασης: 14 Δεκέμβριος 2023. [Έκδοση σε ψηφιακή μορφή]. Διαθέσιμο στο: <https://getsmartgpt.com/>
- [69] ‘Promptist - a Hugging Face Space by microsoft’. Ημερομηνία πρόσβασης: 14 Δεκέμβριος 2023. [Έκδοση σε ψηφιακή μορφή]. Διαθέσιμο στο: <https://huggingface.co/spaces/microsoft/Promptist>
- [70] ‘PromptPerfect - Elevate Your Prompts to Perfection. Prompt Engineering, Optimizing, Debugging and Hosting.’ Ημερομηνία πρόσβασης: 14 Δεκέμβριος 2023. [Έκδοση σε ψηφιακή μορφή]. Διαθέσιμο στο: <https://promptperfect.jina.ai/>

Κυρώσεις για λογοκλοπή

Η λογοκλοπή είναι ένα πολύ σοβαρό παράπτωμα. Με απόφαση της ΓΣΕΣ φοιτητής που διαπιστώνεται ότι υποπίπτει σε λογοκλοπή κατά την εκπόνηση της διπλωματικής του εργασίας αποβάλλεται από το ΠΜΣ. Εάν έχει ήδη αποφοιτήσει ανακαλείται το Μεταπτυχιακό δίπλωμα Ειδίκευσης και προωθείται το θέμα στο Δικαστικό Γραφείο του Πανεπιστημίου για την έναρξη των ανάλογων νομικών διαδικασιών.