



IONIAN UNIVERSITY

Τμήμα Πληροφορικής

- Πτυχιακή εργασία -

**Ανάπτυξη Εικονικού Βοηθού (Virtual Assistant/Chatbot) με
Χρήση της Τεχνολογίας του Υπολογιστικού Νέφους και
Σκοπό την Σύσταση Μουσικής**

Παπαδόπουλος Θεόφιλος Παναγιώτης

Επιβλέπων: Καναβός Ανδρέας

Επιβλέπων

Ανδρέας Καναβός Αναπληρωτής Καθηγητής Ιονίου Πανεπιστήμιου

Τριμελής Επιτροπή

Ανδρέας Καναβός Αναπληρωτής Καθηγητής Ιονίου Πανεπιστήμιου
Κάτια-Λήδα Κερμανίδου Αναπληρώτρια Πρόεδρος Ιονίου Πανεπιστημίου
Αριστείδης Βραχάτης Επίκουρος Καθηγητής Ιονίου Πανεπιστήμιου

ΕΥΧΑΡΙΣΤΙΕΣ

Θέλω να ευχαριστήσω θερμά: Τον επιβλέποντα καθηγητή της πτυχιακής μου εργασίας Δρ Ανδρέα Καναβό για την ευκαιρία, την καθοδήγηση, τις πολύτιμες συμβουλές, ιδέες και υποδείξεις του καθώς και την γενικότερη βοήθεια του σε όλη τη διάρκεια της εκπόνησής της πτυχιακής μου.

ΠΕΡΙΛΗΨΗ

Η παρούσα διατριβή παρουσιάζει την ανάπτυξη ενός συνομιλητικού chatbot που σχεδιάστηκε για να παρέχει εξατομικευμένες μουσικές προτάσεις χρησιμοποιώντας το API του Spotify, με βάση τη συναισθηματική κατάσταση του χρήστη.

Το σύστημα χρησιμοποιεί τεχνικές επεξεργασίας φυσικής γλώσσας (NLP) για την ανάλυση της εισόδου του χρήστη, προσδιορίζοντας τη συναισθηματική διάθεση μέσω ανάλυσης συναισθήματος με τη βοήθεια της βιβλιοθήκης Transformers της Hugging Face. Ενσωματώνει τις δυνατότητες προτάσεων του Spotify για να προτείνει μουσική που ευθυγραμμίζεται με την ανιχνευόμενη διάθεση, προσφέροντας μια απρόσκοπτη και ενδιαφέρουσα εμπειρία χρήστη. Επιπλέον, το chatbot διατηρεί τις προτιμήσεις και τις προηγούμενες αλληλεπιδράσεις του χρήστη για να βελτιώσει την ακρίβεια των προτάσεων με την πάροδο του χρόνου. Μέσω της εφαρμογής μοντέλων μηχανικής μάθησης για την ταξινόμηση προθέσεων και την ανίχνευση διάθεσης, το chatbot δείχνει σημαντικές δυνατότητες στη βελτίωση των συστημάτων εξατομικευμένων μουσικών προτάσεων, συμβάλλοντας στον ευρύτερο τομέα των εφαρμογών AI.

CONTENTS

1. Εισαγωγή.....	1
1.1 Βασικές πληροφορίες.....	1
2. Θεωρητικό Υπόβαθρο.....	3
2.1 ΑΙ.....	3
Ορισμός.....	3
Ιστορία του ΑΙ.....	3
Πλεονεκτήματα προς την κοινωνία.....	6
2.2 Chatbots.....	7
Ιστορία των chatbot.....	8
1. Chatbots Βασισμένα σε Κανόνες.....	11
2. Chatbots με Τεχνητή Νοημοσύνη.....	12
3. Υβριδικά Chatbots.....	13
Τεχνολογίες πίσω από τα Chatbots.....	14
2.2.1 Natural Language Processing.....	14
Προηγμένες Τεχνικές NLP.....	22
Named Entity Recognition (NER).....	22
2.2.1.i Large Language Models.....	23
Τα Μεγάλα Μοντέλα Γλώσσας (LLMs).....	23
2.2.2 Natural Language Understanding.....	25
2.2.3 Διάσημα frameworks για chatbots.....	27
2.2.3.i Rasa.....	27
2.2.3.ii Dialogflow.....	28
2.2.3.iii Landbot AI.....	29
2.2.3.iv TensorFlow.....	30
3. Ανάλυση κωδικα.....	33
3.1 Imports και βιβλιοθήκες.....	36
3.2 Natural Language Processing.....	40
Sentiment analysis.....	44
3.2.1 Τι είναι ένα Pipeline;.....	44
3.2.2 Η Βιβλιοθήκη Transformers από την Hugging Face.....	44
3.2.3 Ανάλυσης Συναισθήματος στο chatbot.....	45
3.3 Spotify API Integration.....	47
Περίληψη API.....	47
3.4 Response.....	56
3.5 ChatSession Class.....	61
Μελλοντικές Βελτιώσεις για το Chatbot και Προβλήματα.....	62
4. Συμπεράσματα και επίλογος.....	65
4.1.1 Λειτουργικότητα και Χρησιμότητα:.....	65

4.1.2 Αλληλεπίδραση Χρήστη:.....	65
4.1.3 Ενσωμάτωση με το API του Spotify:.....	66
4.1.4 NLP και Μοντέλο Μηχανικής Μάθησης:.....	66
4.1.5 Κλιμακωσιμότητα και Ανθεκτικότητα:.....	67
4.1.6 Περιορισμοί και Προκλήσεις:.....	67
Συμπέρασμα.....	67
A. Παράρτημα.....	69
A.1 Βιβλιογραφία και ηλεκτρονικές πηγές.....	69

1. ΕΙΣΑΓΩΓΗ

Ένα chatbot είναι ένα λογισμικό που έχει σχεδιαστεί για να προσομοιώνει την ανθρώπινη συνομιλία. Τα τελευταία χρόνια με τις μεγάλες προόδους τόσο στον ίδιο τον κώδικα των chatbot αλλά και στους πόρους που μπορούμε να τους παρέχουμε, η χρήση τους έχει αυξηθεί εκθετικά σε διάφορους τομείς όπως η εξυπηρέτηση πελατών, η υγειονομική περίθαλψη, τα χρηματοοικονομικά και η ψυχαγωγία. Η ικανότητα των chatbots να παρέχουν υποστήριξη όλο το 24ωρο, να χειρίζονται πολλαπλά ερωτήματα ταυτόχρονα και να προσφέρουν συνεπείς απαντήσεις τα καθιστά ελκυστική λύση για διάφορες επιχειρήσεις που στοχεύουν στη βελτίωση της επικοινωνίας τους με τον πελάτη τους. Ωστόσο, παρά τα πλεονεκτήματα αυτά, πολλά chatbots εξακολουθούν να αντιμετωπίζουν δυσκολίες στο να παρέχουν πραγματικά προσωποποιημένες και συναφείς απαντήσεις, συχνά οδηγώντας σε απογοήτευση των χρηστών. Επίσης αν και πολλά chatbots χρησιμοποιούνται αρκετά σε επαγγελματικούς τομείς ακόμη δεν έχουν καταφέρει να μπουν στην καθημερινότητα του μέσου πολίτη. Αυτό είναι κάτι που αυτή η διατριβή θα δοκιμάσει να κάνει δημιουργώντας έναν “βοηθό” για τους λάτρεις της μουσικής.

1.1 Βασικές πληροφορίες

Αυτή η διατριβή παρουσιάζει την ανάπτυξη ενός συνομιλητικού chatbot που αξιοποιεί Μεγάλα Γλωσσικά Μοντέλα (LLMs) για να προσφέρει προσωποποιημένες αλληλεπιδράσεις και συστάσεις, αντιμετωπίζοντας το προαναφερθέν κενό. Σε αντίθεση με τα παραδοσιακά chatbots που χρησιμοποιούν προκαθορισμένα σενάρια και δέντρα αποφάσεων, τα LLMs μπορούν να κατανοούν και να παράγουν κείμενο

που μοιάζει με την ανθρώπινη ομιλία. Αυτή η δυνατότητα επιτρέπει στο chatbot να κατανοεί τις λεπτομέρειες των ερωτημάτων των χρηστών και να παρέχει πιο ακριβείς και κατάλληλες απαντήσεις.

Το σύστημα χρησιμοποιεί τεχνικές επεξεργασίας φυσικής γλώσσας (NLP) για την ανάλυση της εισόδου του χρήστη, προσδιορίζοντας τη συναισθηματική διάθεση μέσω ανάλυσης συναισθήματος με τη βοήθεια της βιβλιοθήκης Transformers της Hugging Face. Ενσωματώνει τις δυνατότητες προτάσεων του Spotify για να προτείνει μουσική που ευθυγραμμίζεται με την ανιχνευόμενη διάθεση, προσφέροντας μια απρόσκοπτη και ενδιαφέρουσα εμπειρία χρήστη. Επιπλέον, το chatbot αξιοποιεί την εννοιολογική επίγνωση, διατηρώντας τις προτιμήσεις και τις προηγούμενες αλληλεπιδράσεις του χρήστη για να βελτιώσει την ακρίβεια των προτάσεων με την πάροδο του χρόνου. Μέσω της εφαρμογής μοντέλων μηχανικής μάθησης για την ταξινόμηση προθέσεων και την ανίχνευση διάθεσης, το chatbot σαν τεχνολογία δείχνει σημαντικές δυνατότητες στη βελτίωση των συστημάτων εξατομικευμένων μουσικών προτάσεων, συμβάλλοντας στον ευρύτερο τομέα των εφαρμογών AI που γνωρίζουν συναισθήματα.

2. ΘΕΩΡΗΤΙΚΟ ΥΠΟΒΑΘΡΟ

Σε αυτό το κεφάλαιο θα αναφερθούν κάποιες βασικές πληροφορίες για διάφορα κομμάτια όπως το AI και η ιστορία του, τι είναι ένα chatbot και πως φτάσαμε στο σήμερα και πληροφορίες για το πως λειτουργούν οι τεχνολογίες πίσω από αυτά.

2.1 AI

Ορισμός

Η Τεχνητή Νοημοσύνη (TN) είναι ένας τομέας της πληροφορικής που επικεντρώνεται στη δημιουργία συστημάτων και αλγορίθμων που μπορούν να εκτελούν καθήκοντα τα οποία απαιτούν ανθρώπινη νοημοσύνη. Αυτά τα καθήκοντα περιλαμβάνουν, μεταξύ άλλων, την εκμάθηση από δεδομένα (machine learning), την κατανόηση και επεξεργασία φυσικής γλώσσας (natural language processing), τη λήψη αποφάσεων και την επίλυση προβλημάτων.[1] Η TN επιδιώκει να προσομοιώσει την ανθρώπινη σκέψη και λογική, επιτρέποντας στα μηχανήματα να μαθαίνουν και να προσαρμόζονται αυτόνομα σε νέες καταστάσεις.

Ιστορία του AI

Η ιστορία της τεχνητής νοημοσύνης (AI) είναι μία πολύ πλούσια ακολουθία προόδων

και καινοτομιών που εκτείνεται σχεδόν έναν αιώνα. Τα θεμέλια του AI τέθηκαν το 1936, όταν ο Βρετανός μαθηματικός Άλαν Τιούρινγκ εισήγαγε την έννοια της «μηχανής Turing». Αυτή η θεωρητική συσκευή μπορούσε να προσομοιώσει οποιονδήποτε υπολογιστικό αλγόριθμο, θέτοντας τα θεμέλια για τον σύγχρονο υπολογισμό και του AI [2]. Η επίσημη “γέννηση” του AI ως πεδίο σημειώθηκε το 1956, όταν ο John McCarthy επινόησε τον όρο «τεχνητή νοημοσύνη» κατά τη διάρκεια του συνεδρίου Dartmouth. Σε αυτό το συνέδριο παρουσιάστηκε και ο Logic Theorist, το πρώτο πρόγραμμα AI [3].

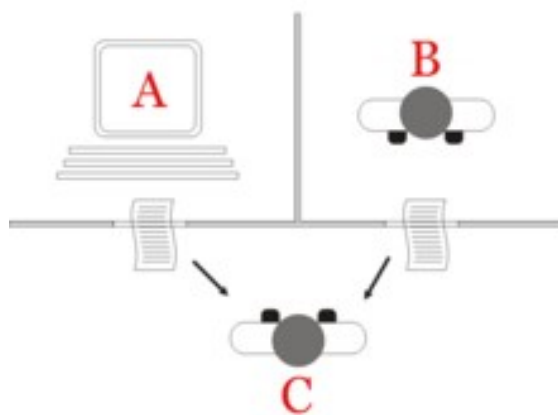


Figure 1. Turing Test, Evaluation of a chatbot [4]

Η περίοδος από τη δεκαετία του 1960 έως τη δεκαετία του 1980 γνώρισε πρώιμες καινοτομίες στο AI. Το 1966, ο Joseph Weizenbaum ανέπτυξε την ELIZA, το πρώτο chatbot ικανό να προσομοιώνει μια συνομιλία με έναν άνθρωπο χρησιμοποιώντας απλά σενάρια [5]. Ένα άλλο σημαντικό ορόσημο ήταν η ανάπτυξη του MYCIN το 1972, ενός ειδικού συστήματος που δημιουργήθηκε στο Πανεπιστήμιο Stanford και έδειξε τις δυνατότητες του AI στον ιατρικό τομέα διαγνώσκοντας βακτηριακές λοιμώξεις και προτείνοντας θεραπείες [6]. Το 1986, η δημιουργία του NETtalk, ενός πρώιμου νευρωνικού δικτύου, αποτέλεσε σημαντική πρόοδο καθώς μπορούσε να μάθει να διαβάζει και να προφέρει λέξεις εκπαιδευόμενο σε φωνητικά δεδομένα[7].

```

Welcome to
      EEEEE LL      IIII ZZZZZZZZ AAAAA
      EE   LL      II   ZZ   AA  AA
      EEEEE LL      II   ZZZ  AAAAAA
      EE   LL      II   ZZ   AA  AA
      EEEEE LLLLLL IIII ZZZZZZZZ AA  AA

Eliza is a mock Rogerian psychotherapist.
The original program was described by Joseph Weizenbaum in 1966.
This implementation by Norbert Landsteiner 2005.

ELIZA: Is something troubling you ?
YOU:   Men are all alike.
ELIZA: What is the connection, do you suppose ?
YOU:   They're always bugging us about something or other.
ELIZA: Can you think of a specific example ?

```

Figure 2. Eliza chatbot, simple conversation [8]

Η δεκαετία του 1990 και η δεκαετία του 2000 έφεραν σημαντικά επιτεύγματα στο ΑΙ. Το 1997, το Deep Blue της IBM νίκησε τον παγκόσμιο πρωταθλητή σκακιού Garry Kasparov, δείχνοντας τη δύναμη του ΑΙ στα στρατηγικά παιχνίδια [9]. Μέχρι το 2011, το ΑΙ άρχισε να μπαίνει πιο εμφανώς στην καθημερινή ζωή με εφαρμογές όπως η Siri της Apple και το Watson της IBM, το οποίο κέρδισε στο τηλεπαιχνίδι Jeopardy! κατανοώντας και απαντώντας σε πολύπλοκες ερωτήσεις [10].

Στη σύγχρονη εποχή, από τη δεκαετία του 2010 έως σήμερα, οι τεχνολογίες ΑΙ έχουν προοδεύσει ραγδαία, λόγω βελτιώσεων σε hardware, τη διαθεσιμότητα δεδομένων και τις αλγοριθμικές καινοτομίες. Οι εφαρμογές του ΑΙ κυμαίνονται πλέον από φωνητικούς βοηθούς και αυτόνομα οχήματα έως και σύνθετη ανάλυση δεδομένων και προγνωστική μοντελοποίηση σε διάφορες βιομηχανίες. Αυτές οι πρόοδοι αντανakλούν την αυξανόμενη επιρροή και τις ικανότητες του ΑΙ, υπογραμμίζοντας τον σημαντικό της ρόλο στη σύγχρονη κοινωνία και τις δυνατότητες για μελλοντικές εξελίξεις.

Πλεονεκτήματα προς την κοινωνία

Η τεχνητή νοημοσύνη (AI) προσφέρει σημαντικά οφέλη στην κοινωνία, συμβάλλοντας σε διάφορους τομείς και βελτιώνοντας την ποιότητα ζωής. Ένα από τα βασικά πλεονεκτήματα είναι η αύξηση της αποδοτικότητας και της παραγωγικότητας σε πολλούς τομείς. Μέσω της αυτοματοποίησης επαναλαμβανόμενων και χρονοβόρων εργασιών, το AI επιτρέπει στους εργαζόμενους να επικεντρωθούν σε πιο δημιουργικές και στρατηγικές δραστηριότητες.

Στον τομέα της υγείας, το AI έχει οδηγήσει σε σημαντικές προόδους, βελτιώνοντας τη διάγνωση και τη θεραπεία ασθενειών. Τα συστήματα AI μπορούν να αναλύουν μεγάλα σύνολα δεδομένων υγείας για να εντοπίζουν μοτίβα και να προβλέπουν πιθανές επιδημίες ή να προσφέρουν εξατομικευμένες θεραπείες. Ένα παράδειγμα είναι η χρήση του AI για την ανάλυση ακτινογραφιών, η οποία μπορεί να ανιχνεύσει καρκινικές αλλαγές με μεγαλύτερη ακρίβεια από τις παραδοσιακές μεθόδους.



Figure 3. Different areas AI can help in Health [11]

Στον τομέα της εκπαίδευσης, το AI συμβάλλει στη δημιουργία προσαρμοσμένων μαθησιακών εμπειριών για τους μαθητές. Τα έξυπνα συστήματα μπορούν να προσαρμόζουν το εκπαιδευτικό υλικό και τις δραστηριότητες με βάση τις ανάγκες και τις ικανότητες κάθε μαθητή, βελτιώνοντας την εκπαιδευτική απόδοση και την αφομοίωση των γνώσεων[12].

Το AI παίζει επίσης καθοριστικό ρόλο στην προώθηση της βιώσιμης ανάπτυξης και της προστασίας του περιβάλλοντος. Μέσω της ανάλυσης δεδομένων και της προγνωστικής μοντελοποίησης, οι τεχνολογίες AI μπορούν να βοηθήσουν στην αποδοτικότερη διαχείριση των φυσικών πόρων και στη μείωση των εκπομπών ρύπων. Επιπλέον, χρησιμοποιούνται για την παρακολούθηση και την αντιμετώπιση της κλιματικής αλλαγής, προβλέποντας περιβαλλοντικές μεταβολές και βοηθώντας στη λήψη προληπτικών μέτρων[13].

Τέλος, το AI βελτιώνει την ασφάλεια και την ποιότητα ζωής στις πόλεις μέσω των «έξυπνων πόλεων». Οι εφαρμογές AI μπορούν να βελτιστοποιούν την κυκλοφορία, να διαχειρίζονται την κατανάλωση ενέργειας και να ενισχύουν την ασφάλεια μέσω προγνωστικών μοντέλων εγκληματικότητας και επιτήρησης. Αυτή οι τεχνολογίες συμβάλλουν στην δημιουργία βιώσιμων και ευφών αστικών κέντρων.

2.2 Chatbots

Ένα chatbot είναι ένα πρόγραμμα υπολογιστή σχεδιασμένο να προσομοιώνει την ανθρώπινη συνομιλία μέσω αλληλεπιδράσεων κειμένου ή φωνής. Χρησιμοποιώντας επεξεργασία φυσικής γλώσσας (NLP) και αλγόριθμους μηχανικής μάθησης, τα chatbots ερμηνεύουν την είσοδο του χρήστη και παρέχουν κατάλληλες απαντήσεις, επιτρέποντας αυτοματοποιημένη εξυπηρέτηση πελατών, προσωπική βοήθεια και ανάκτηση πληροφοριών. Μπορούν να ενσωματωθούν σε διάφορες πλατφόρμες, όπως ιστοσελίδες, εφαρμογές μηνυμάτων και εικονικούς βοηθούς, για να βελτιώσουν την εμπειρία χρήστη και να διευκολύνουν τις διαδικασίες επικοινωνίας. Επιπλέον, τα

chatbots μπορούν να χρησιμοποιηθούν για την αυτοματοποίηση επιχειρησιακών διαδικασιών, την παροχή εξατομικευμένων προτάσεων και την ανάλυση δεδομένων χρήστη, καθιστώντας τα ένα πολύτιμο εργαλείο σε διάφορους τομείς όπως το λιανεμπόριο, η υγειονομική περίθαλψη, η τραπεζική και η εκπαίδευση. [14]

Ιστορία των chatbot

Η ιστορία των chatbots είναι μια συναρπαστική πορεία τεχνολογικής εξέλιξης που ξεκινά από τις αρχές της πληροφορικής μέχρι τις σύγχρονες εφαρμογές Τεχνητής Νοημοσύνης (TN). Ακολουθεί μια λεπτομερής αναδρομή στην ανάπτυξη των chatbots, από την ιδέα τους έως την τρέχουσα κατάσταση, με έμφαση στην ακρίβεια και την εξέλιξή τους.

1. Η Ιδέα των Chatbots

Η ιδέα των chatbots γεννήθηκε από την επιθυμία να δημιουργηθούν μηχανές που μπορούν να αλληλεπιδρούν με ανθρώπους χρησιμοποιώντας φυσική γλώσσα. Η αρχική σκέψη ήταν να αναπτυχθούν προγράμματα που θα μπορούσαν να κατανοούν και να ανταποκρίνονται σε ανθρώπινες ερωτήσεις και εντολές, προσομοιώνοντας τη συνομιλία.

Το Πρώτο Chatbot: ELIZA

Το πρώτο chatbot όπως προαναφέρθηκε, γνωστό ως ELIZA, δημιουργήθηκε από τον Joseph Weizenbaum το 1966 στο MIT. Το ELIZA σχεδιάστηκε για να προσομοιώνει έναν ψυχοθεραπευτή, χρησιμοποιώντας μια απλή τεχνική ανάλυσης κειμένου που αναγνωρίζει λέξεις-κλειδιά και παράγει προκαθορισμένες απαντήσεις. Παρά την περιορισμένη τεχνολογία της εποχής, το ELIZA κατάφερε να ξεγελάσει αρκετούς ανθρώπους, κάνοντάς τους να πιστεύουν ότι συνομιλούσαν με έναν πραγματικό άνθρωπο [15].



Figure 4. A conversation between a human and ELIZA's DOCTOR script [16]

2. Η Εξέλιξη των Chatbots

1970s-1980s: Early Development

Κατά τη δεκαετία του 1970 και 1980, η τεχνολογία των chatbots προχώρησε σχετικά πιο αργά, με την ανάπτυξη πιο προηγμένων συστημάτων όπως το PARRY, το οποίο προσομοίωνε έναν σχιζοφρενή ασθενή. Τα chatbots αυτής της περιόδου βασίζονταν κυρίως σε απλούς κανόνες και περιορισμένες βάσεις δεδομένων.

1990s: The Internet Era

Με την ανάπτυξη του Διαδικτύου τη δεκαετία του 1990, τα chatbots άρχισαν να γίνονται πιο διαδραστικά και προσβάσιμα. Ένα χαρακτηριστικό παράδειγμα είναι το ALICE (Artificial Linguistic Internet Computer Entity), που αναπτύχθηκε από τον Richard Wallace το 1995. Το ALICE χρησιμοποίησε την AIML (Artificial Intelligence Markup Language) για να δημιουργήσει πιο σύνθετες και ρεαλιστικές συνομιλίες, αν και ακόμα περιοριζόταν από την τεχνολογία της εποχής [17].

2000s: Machine Learning and NLP

Η επόμενη μεγάλη εξέλιξη ήρθε με την πρόοδο στη μηχανική μάθηση (machine learning) και την επεξεργασία φυσικής γλώσσας (NLP). Τα chatbots άρχισαν να χρησιμοποιούν τεχνικές στατιστικής ανάλυσης και μεγάλα σύνολα δεδομένων για να βελτιώσουν την ακρίβεια και την αποτελεσματικότητά τους. Παραδείγματα αυτής της περιόδου περιλαμβάνουν το Cleverbot, που δημιουργήθηκε το 2006 και χρησιμοποιεί αλγόριθμους μάθησης για να βελτιώνει τις απαντήσεις του με την πάροδο του χρόνου .

NLP Pipeline

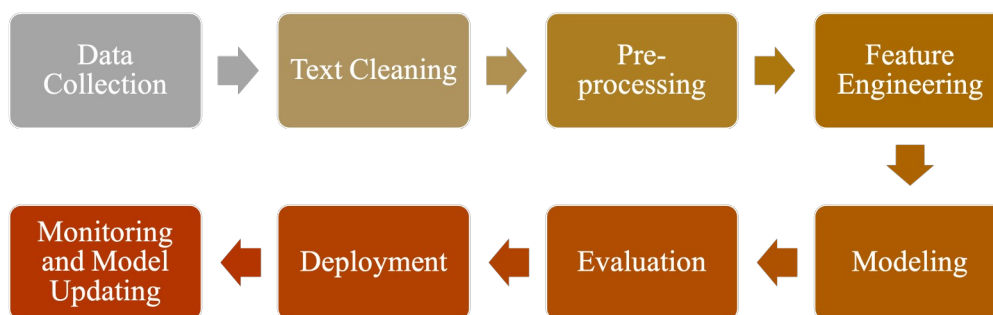


Figure 5. NLP Pipeline [18]

2010s: The Rise of AI-Powered Chatbots

Η δεκαετία του 2010 είδε μια επανάσταση στα chatbots με την έλευση των μεγάλων γλωσσικών μοντέλων (LLMs) και της βαθιάς μάθησης (deep learning). Οι σύγχρονοι βοηθοί όπως το Siri της Apple, το Alexa της Amazon και το Google Assistant χρησιμοποιούν τεχνολογίες TN για να κατανοούν και να ανταποκρίνονται σε φωνητικές εντολές, παρέχοντας εξαιρετικά ακριβείς και χρήσιμες απαντήσεις. [19]

Σύγχρονες Εφαρμογές και Μελλοντικές Προοπτικές

Σήμερα, τα chatbots χρησιμοποιούνται ευρέως σε πολλούς τομείς, από την εξυπηρέτηση πελατών και την εκπαίδευση έως την υγειονομική περίθαλψη και την ψυχαγωγία. Με τη συνεχή βελτίωση των αλγορίθμων TN και την πρόσβαση σε τεράστια σύνολα δεδομένων, τα chatbots γίνονται ολοένα και πιο ικανά να παρέχουν εξατομικευμένες και χρήσιμες απαντήσεις σε πραγματικό χρόνο.

3. Κατηγοριοποίηση των Chatbots

Τα chatbots μπορούν να κατηγοριοποιηθούν γενικά με βάση τη λειτουργικότητά τους και την υποκείμενη τεχνολογία που καθοδηγεί τις απαντήσεις τους. Η κατανόηση αυτών των κατηγοριών βοηθά στην κατανόηση των διαφορετικών δυνατοτήτων και εφαρμογών των chatbots. Οι κύριοι τύποι περιλαμβάνουν τα chatbots βασισμένα σε κανόνες, τα chatbots που χρησιμοποιούν τεχνητή νοημοσύνη και τα υβριδικά chatbots. Αυτή η ανάλυση εξετάζει κάθε τύπο, διερευνώντας τους μηχανισμούς λειτουργίας τους, τις διαφορετικές προσεγγίσεις και τις τεχνολογίες που χρησιμοποιούν.

1. Chatbots Βασισμένα σε Κανόνες

Τα chatbots βασισμένα σε κανόνες, επίσης γνωστά ως script chatbots, λειτουργούν με ένα προκαθορισμένο σύνολο κανόνων και δέντρων αποφάσεων. Αυτοί οι κανόνες δημιουργούνται από προγραμματιστές ή ειδικούς του τομέα και καθορίζουν πώς το chatbot θα πρέπει να απαντά σε συγκεκριμένες εισόδους χρηστών. Τα chatbots βασισμένα σε κανόνες συνήθως ακολουθούν ένα δένδροειδές μοντέλο αποφάσεων, όπου κάθε κόμβος αντιπροσωπεύει μια είσοδο χρήστη και οι κλάδοι αντιπροσωπεύουν πιθανές απαντήσεις. Επίσης, χρησιμοποιούν τεχνικές αντιστοίχισης προτύπων, όπου το bot αναζητά συγκεκριμένες λέξεις-κλειδιά ή φράσεις στην είσοδο του χρήστη και απαντά ανάλογα. Επιπλέον, αυτά τα chatbots εφαρμόζουν μια σειρά από συνθήκες (IF-THEN) για να καθορίσουν την κατάλληλη απάντηση βάσει της εισόδου του χρήστη. [20]

Τα κύρια πλεονεκτήματα των chatbots βασισμένων σε κανόνες περιλαμβάνουν την ευκολία σχεδίασης και υλοποίησης, προβλέψιμες απαντήσεις και αποτελεσματικότητα για απλές ερωτήσεις και ευθείς εργασίες. Ωστόσο, περιορίζονται στην ικανότητά τους να κατανοούν εισόδους πέρα από τους προκαθορισμένους κανόνες, δυσκολεύονται να χειριστούν περίπλοκες ή ποικίλες εισόδους χρηστών και απαιτούν εκτεταμένες χειροκίνητες ενημερώσεις για να διατηρήσουν τη λειτουργικότητά τους. Φυσικά, το αρνητικό τους είναι προφανές, καθώς είναι αρκετά περιορισμένα στις απαντήσεις που

μπορούν να δώσουν, δεδομένου ότι όλα είναι προκαθορισμένα από τον δημιουργό του chatbot.

2. Chatbots με Τεχνητή Νοημοσύνη

Τα chatbots με τεχνητή νοημοσύνη χρησιμοποιούν διάφορες διαθέσιμες τεχνικές τεχνητής νοημοσύνης και μηχανικής μάθησης για να κατανοούν και να απαντούν στις εισόδους των χρηστών. Αυτά τα chatbots μπορούν να μάθουν από τις αλληλεπιδράσεις και να βελτιώνονται με την πάροδο του χρόνου, καθιστώντας τα πιο ευέλικτα και ικανά να χειρίζονται σύνθετες συνομιλίες. Χρησιμοποιούν τεχνικές επεξεργασίας φυσικής γλώσσας (NLP) για να κατανοούν και να επεξεργάζονται την ανθρώπινη γλώσσα. Βασικές εργασίες του NLP περιλαμβάνουν την ανάλυση τοκενιζαρισμένων δεδομένων, την αναγνώριση μερών του λόγου, την αναγνώριση οντοτήτων και την ανάλυση συναισθημάτων. Αλγόριθμοι μηχανικής μάθησης, όπως τα νευρωνικά δίκτυα, τα δέντρα αποφάσεων και οι υποστηρικτές διανύσματος, χρησιμοποιούνται για να μαθαίνουν από δεδομένα και να κάνουν προβλέψεις. Τα μοντέλα βαθιάς μάθησης, όπως τα συνελκτικά νευρωνικά δίκτυα (CNNs)[[21](#)] και τα αναδρομικά νευρωνικά δίκτυα (RNNs)[[22](#)], επεξεργάζονται και κατανοούν μεγάλα ποσά μη δομημένων δεδομένων. Τα chatbots με τεχνητή νοημοσύνη διατηρούν επίσης το πλαίσιο στις συνομιλίες, επιτρέποντάς τους να χειρίζονται πολυγυριστικούς διαλόγους αποτελεσματικά.

Παραδείγματα chatbots με τεχνητή νοημοσύνη περιλαμβάνουν εικονικούς βοηθούς όπως η Siri της Apple, η Alexa της Amazon και το Google Assistant, που χρησιμοποιούν την τεχνητή νοημοσύνη για να εκτελούν μια ευρεία γκάμα εργασιών. Τα bots εξυπηρέτησης πελατών, όπως αυτά που προσφέρονται από το IBM Watson και το LivePerson, χρησιμοποιούν την τεχνητή νοημοσύνη για να κατανοούν και να επιλύουν σύνθετες ερωτήσεις. Τα πλεονεκτήματα των chatbots με τεχνητή νοημοσύνη περιλαμβάνουν βελτιωμένη κατανόηση και χειρισμό σύνθετων ερωτήσεων, την ικανότητα να μαθαίνουν και να βελτιώνονται με την πάροδο του χρόνου και αλληλεπιδράσεις που γνωρίζουν το πλαίσιο. Ωστόσο, απαιτούν

σημαντικά ποσά δεδομένων για εκπαίδευση, είναι υπολογιστικά έντονα και μπορεί να έχουν προκαταλήψεις στα μοντέλα τους .

3. Υβριδικά Chatbots

Τα υβριδικά chatbots συνδυάζουν την απλότητα των συστημάτων βασισμένων σε κανόνες με την πολυπλοκότητα της τεχνητής νοημοσύνης. Χρησιμοποιούν προσεγγίσεις βασισμένες σε κανόνες για την αντιμετώπιση απλών εργασιών και τεχνικές τεχνητής νοημοσύνης για πιο σύνθετες αλληλεπιδράσεις. Τα υβριδικά chatbots συχνά υλοποιούν μηχανισμούς εφεδρείας, όπου το στοιχείο της τεχνητής νοημοσύνης επανέρχεται στις απαντήσεις βασισμένες σε κανόνες εάν δεν καταφέρει να κατανοήσει μια ερώτηση. Χρησιμοποιούν επίσης μια αρχιτεκτονική με επίπεδα, όπου το πρώτο επίπεδο χειρίζεται απλές ερωτήσεις μέσω κανόνων και τα επόμενα επίπεδα χρησιμοποιούν την τεχνητή νοημοσύνη για πιο σύνθετες ερωτήσεις . Η εναλλαγή πλαισίου είναι ένα άλλο χαρακτηριστικό των υβριδικών chatbots, επιτρέποντας την ομαλή εναλλαγή μεταξύ απαντήσεων βασισμένων σε κανόνες και τεχνητή νοημοσύνη ανάλογα με την πολυπλοκότητα της εισόδου του χρήστη .

Παραδείγματα υβριδικών chatbots περιλαμβάνουν bots ηλεκτρονικού εμπορίου που χειρίζονται βασικές ερωτήσεις όπως η κατάσταση παραγγελίας μέσω κανόνων, ενώ χρησιμοποιούν την τεχνητή νοημοσύνη για να παρέχουν εξατομικευμένες προτάσεις προϊόντων . Τα bots υγείας διαχειρίζονται τον προγραμματισμό ραντεβού μέσω κανόνων και χρησιμοποιούν την τεχνητή νοημοσύνη για να προσφέρουν ιατρικές συμβουλές βάσει των συμπτωμάτων του χρήστη . Τα πλεονεκτήματα των υβριδικών chatbots περιλαμβάνουν την ισορροπία μεταξύ αποδοτικότητας και πολυπλοκότητας, πιο ακριβείς και σχετικές απαντήσεις και την προσαρμοστικότητα σε διάφορες περιπτώσεις χρήσης. Ωστόσο, είναι περίπλοκα στο σχεδιασμό και την υλοποίηση και απαιτούν προσεκτική ενσωμάτωση των στοιχείων βασισμένων σε κανόνες και τεχνητή νοημοσύνη .

Τεχνολογίες πίσω από τα Chatbots

Η επεξεργασία φυσικής γλώσσας (NLP) είναι μια κρίσιμη τεχνολογία για τα chatbots, επιτρέποντάς τους να κατανοούν και να παράγουν ανθρώπινη γλώσσα. Βασικές εργασίες του NLP περιλαμβάνουν την τοκενιζαρισμένη ανάλυση (tokenization) (διάσπαση του κειμένου σε μεμονωμένες λέξεις ή τοκέν), την αναγνώριση μερών του λόγου (αναγνώριση των γραμματικών μερών του λόγου σε μια πρόταση), την αναγνώριση οντοτήτων (ανίχνευση και ταξινόμηση οντοτήτων όπως ονόματα, ημερομηνίες και τοποθεσίες) και την ανάλυση συναισθημάτων (προσδιορισμός του συναισθήματος που εκφράζεται σε ένα κείμενο, όπως θετικό, αρνητικό ή ουδέτερο). [\[23\]](#)

Οι τεχνολογίες μηχανικής μάθησης και βαθιάς μάθησης δίνουν την δυνατότητα στα chatbots που χρησιμοποιούν τεχνητή νοημοσύνη να μαθαίνουν από δεδομένα και να βελτιώνουν την απόδοσή τους. Η επιβλεπόμενη μάθηση περιλαμβάνει την εκπαίδευση μοντέλων σε επισημασμένα δεδομένα για να προβλέπουν αποτελέσματα για νέες εισόδους, ενώ η μη επιβλεπόμενη μάθηση εντοπίζει μοτίβα στα δεδομένα χωρίς επισημασμένα αποτελέσματα. Τα νευρωνικά δίκτυα, συμπεριλαμβανομένων των μοντέλων βαθιάς μάθησης όπως τα CNNs και τα RNNs, χρησιμοποιούνται για την επεξεργασία και κατανόηση σύνθετων δεδομένων.

Τα chatbots συχνά χρησιμοποιούν APIs για να επεκτείνουν τις δυνατότητές τους. Η ενσωμάτωση με πλατφόρμες μηνυμάτων όπως το Facebook Messenger, το Slack και το WhatsApp επιτρέπει στα chatbots να αλληλεπιδρούν με χρήστες σε δημοφιλείς πλατφόρμες μηνυμάτων. APIs από υπηρεσίες τρίτων όπως το Google Cloud NLP, το IBM Watson και το Microsoft Azure παρέχουν προηγμένες δυνατότητες NLP και τεχνητής νοημοσύνης.

2.2.1 Natural Language Processing

Η Επεξεργασία Φυσικής Γλώσσας (ΕΦΓ) ή Natural Language Processing (NLP) είναι ένας τομέας της τεχνητής νοημοσύνης που ασχολείται με την αλληλεπίδραση μεταξύ των υπολογιστών και της ανθρώπινης γλώσσας. Ο στόχος της NLP είναι να επιτρέψει

στους υπολογιστές να κατανοούν, να ερμηνεύουν και να δημιουργούν ανθρώπινη γλώσσα με τρόπους που είναι χρήσιμοι. Αυτό περιλαμβάνει την ανάλυση, την κατανόηση και τη δημιουργία κειμένου και ομιλίας σε φυσικές γλώσσες όπως τα αγγλικά, τα ελληνικά, τα κινέζικα και άλλες. [24]

Νευρώνες και Νευρωνικά Δίκτυα

Οι νευρώνες είναι οι βασικές μονάδες των νευρωνικών δικτύων. Κάθε νευρώνας λαμβάνει μία ή περισσότερες εισόδους, τις επεξεργάζεται μέσω μιας ενεργοποιητικής συνάρτησης, και παράγει μία έξοδο. Οι νευρώνες οργανώνονται σε στρώματα (layers), με τα δεδομένα να περνούν από το ένα στρώμα στο άλλο.

Στα feedforward νευρωνικά δίκτυα, τα δεδομένα κινούνται μόνο προς μία κατεύθυνση, από την είσοδο προς την έξοδο, χωρίς να υπάρχουν κυκλικές συνδέσεις. Τα deep learning μοντέλα είναι feedforward νευρωνικά δίκτυα με πολλαπλά κρυφά στρώματα (hidden layers).

Ενεργοποιητική Συνάρτηση ReLU

Η ReLU (Rectified Linear Unit) είναι μία από τις πιο δημοφιλείς ενεργοποιητικές συναρτήσεις στα νευρωνικά δίκτυα. Ορίζεται ως:

$$\text{ReLU}(x) = \max(0, x)$$

Αυτό σημαίνει ότι επιστρέφει το ίδιο το εισερχόμενο σήμα αν είναι θετικό, αλλιώς επιστρέφει μηδέν. Η ReLU βοηθά στην επίλυση του προβλήματος της εξαφάνισης των gradients (vanishing gradients) που μπορεί να συμβεί με άλλες ενεργοποιητικές συναρτήσεις όπως η sigmoid ή η tanh. [25]

Epoch/Εποχή

Μία epoch/εποχή αναφέρεται σε μία πλήρη διέλευση όλων των δειγμάτων στο σύνολο εκπαίδευσης μέσω του νευρωνικού δικτύου. Κατά τη διάρκεια της εκπαίδευσης, το μοντέλο αναπροσαρμόζει τα βάρη του πολλές φορές, περνώντας από

τα δεδομένα εκπαίδευσης σε πολλές epochs για να μειώσει το σφάλμα και να βελτιώσει την απόδοση. [26]

Βασικές πληροφορίες και ιστορία της NLP

Η NLP έχει μια μακρά ιστορία που ξεκινά από τις πρώτες προσπάθειες να μεταφράζονται αυτόματα κείμενα μεταξύ γλωσσών κατά τη διάρκεια του Ψυχρού Πολέμου. Με την ανάπτυξη των υπολογιστών και την πρόοδο της τεχνητής νοημοσύνης, η NLP εξελίχθηκε για να περιλαμβάνει ένα ευρύ φάσμα εφαρμογών όπως η αυτόματη μετάφραση, οι συστήματα συνομιλίας, η ανάλυση συναισθήματος και πολλά άλλα.

Κύριες Τεχνικές και Μέθοδοι

1. Στατιστικές Μέθοδοι και Μηχανική Μάθηση: Οι πρώτες προσεγγίσεις στην NLP βασίζονταν σε στατιστικές μεθόδους και μοντέλα μηχανικής μάθησης που εκπαιδεύονταν σε μεγάλες ποσότητες κειμένου. Αυτά τα μοντέλα χρησιμοποιούνταν για να προβλέψουν τις πιθανότητες εμφάνισης λέξεων ή φράσεων.
2. Νευρωνικά Δίκτυα και Βαθιά Μάθηση: Με την πρόοδο της βαθιάς μάθησης, τα νευρωνικά δίκτυα άρχισαν να χρησιμοποιούνται ευρέως στην NLP. Τα μοντέλα όπως τα RNN (Recurrent Neural Networks) και τα LSTM (Long Short-Term Memory) αποδείχτηκαν ιδιαίτερα αποτελεσματικά στην επεξεργασία ακολουθιών δεδομένων όπως το κείμενο.
3. Μετασχηματιστές και BERT: Μια σημαντική πρόοδος στην NLP ήταν η ανάπτυξη του μοντέλου μετασχηματιστών, με το BERT (Bidirectional Encoder Representations from Transformers) να είναι ένα από τα πιο γνωστά παραδείγματα. Αυτά τα μοντέλα έχουν επιτύχει αξιοσημείωτη ακρίβεια σε πολλές εργασίες NLP, όπως η κατανόηση ερωτήσεων, η αναγνώριση οντοτήτων και η παραγωγή κειμένου.

Εφαρμογές NLP

Η NLP έχει πολλές πρακτικές εφαρμογές που χρησιμοποιούνται καθημερινά:

1. **Αυτόματη Μετάφραση:** Εργαλεία όπως το Google Translate χρησιμοποιούν NLP για να μεταφράζουν κείμενο από μια γλώσσα σε άλλη.
2. **Συστήματα Συνομιλίας και Chatbots:** Τα chatbots χρησιμοποιούν NLP για να κατανοούν και να απαντούν σε ερωτήσεις χρηστών με φυσικό τρόπο.
3. **Ανάλυση Συναισθήματος:** Χρησιμοποιείται για να καθορίζει την συναισθηματική χροιά σε κείμενα, όπως οι αξιολογήσεις προϊόντων ή τα σχόλια στα μέσα κοινωνικής δικτύωσης.
4. **Κατηγοριοποίηση Κειμένου:** Χρησιμοποιείται για την ομαδοποίηση και την κατάταξη κειμένων σε κατηγορίες, όπως η ταξινόμηση email σε εισερχόμενα ή spam.

Πληροφορίες για NLP στον Κώδικα

Στον κώδικα του chatbot, το νευρωνικό δίκτυο χρησιμοποιείται για την ανάλυση συναισθημάτων και την κατηγοριοποίηση της διάθεσης. Τα βήματα υλοποίησης είναι τα εξής:

Προετοιμασία των Δεδομένων

Το πρώτο βήμα στην εκπαίδευση ενός νευρωνικού δικτύου είναι η προετοιμασία του συνόλου δεδομένων. Για το chatbot, το σύνολο δεδομένων αποτελείται από διάφορες προθέσεις, καθεμία από τις οποίες περιέχει μοτίβα (παραδείγματα φράσεων) και τις αντίστοιχες απαντήσεις. Αυτά τα δεδομένα αποθηκεύονται σε ένα αρχείο JSON, το οποίο φορτώνεται και προεπεξεργάζεται για να δημιουργήσει μια κατάλληλη μορφή για την εκπαίδευση του μοντέλου.

- **Φόρτωση Δεδομένων:** Οι προθέσεις φορτώνονται από ένα αρχείο JSON σε ένα λεξικό Python. Τα intents είναι προτάσεις όπως "Ποιο είναι το όνομά σου" για τις οποίες θα θέλαμε να υπάρχουν πιο συγκεκριμένες απαντήσεις που δεν θα μπορούσε να βρει κάπου το chatbot.
- **Διαχωρισμός και Λημματοποίηση:** Κάθε μοτίβο στις προθέσεις διαχωρίζεται και λημματοποιείται. Ο διαχωρισμός διαχωρίζει το κείμενο σε μεμονωμένες λέξεις, ενώ η λημματοποίηση μειώνει κάθε λέξη στη βασική της μορφή. Αυτή η προεπεξεργασία βοηθά στην κανονικοποίηση των δεδομένων κειμένου.
- **Δημιουργία του Bag of Words (BoW):** Οι διαχωρισμένες και λημματοποιημένες λέξεις μετατρέπονται σε αναπαράσταση BoW. Αυτό περιλαμβάνει τη δημιουργία ενός δυαδικού διανύσματος για κάθε μοτίβο, που υποδεικνύει την παρουσία ή απουσία κάθε λέξης στο λεξιλόγιο.

The following models a text document using bag-of-words. Here are two simple text documents:

(1) John likes to watch movies. Mary likes movies too.

(2) Mary also likes to watch football games.

Based on these two text documents, a list is constructed as follows for each document:

"John", "likes", "to", "watch", "movies", "Mary", "likes", "movies", "too"
 "Mary", "also", "likes", "to", "watch", "football", "games"

Representing each bag-of-words as a **JSON object**, and attributing to the respective **JavaScript** variable:

```
BoW1 = {"John":1,"likes":2,"to":1,"watch":1,"movies":2,"Mary":1,"too":1};
BoW2 = {"Mary":1,"also":1,"likes":1,"to":1,"watch":1,"football":1,"games":1};
```

Figure 6. Bag Of Words[27]

Αρχιτεκτονική Νευρωνικού Δικτύου

Το νευρωνικό δίκτυο υλοποιείται χρησιμοποιώντας το TensorFlow και το Keras. Το μοντέλο αποτελείται από μια είσοδο, δύο κρυφά επίπεδα με ReLU ενεργοποιήσεις και dropout για κανονικοποίηση, και ένα επίπεδο εξόδου με softmax ενεργοποίηση.

- **Επίπεδο Εισόδου:** Το μέγεθος του επιπέδου εισόδου καθορίζεται από το μήκος της αναπαράστασης BoW.
- **Κρυφά Επίπεδα:** Δύο πλήρη επίπεδα με ReLU ενεργοποίηση και dropout για κανονικοποίηση. Το dropout βοηθά στην αποτροπή υπερπροσαρμογής, ρυθμίζοντας τυχαία ένα ποσοστό των μονάδων εισόδου σε 0 σε κάθε ενημέρωση κατά τη διάρκεια της εκπαίδευσης.
- **Επίπεδο Εξόδου:** Ένα πλήρες επίπεδο με softmax ενεργοποίηση, που παράγει μια κατανομή πιθανότητας στις τάξεις (προθέσεις).

Παράμετροι Εκπαίδευσης

- **Συνάρτηση Απώλειας:** Το μοντέλο χρησιμοποιεί την κατηγορική διασταυρούμενη εντροπία ως συνάρτηση απώλειας, κατάλληλη για προβλήματα πολυκατηγορικής ταξινόμησης. Η κατηγορική διασταυρούμενη εντροπία μετρά τη διαφορά μεταξύ των πραγματικών ετικετών και των προβλεπόμενων πιθανοτήτων, βοηθώντας το μοντέλο να προσαρμοστεί ώστε να κάνει ακριβέστερες προβλέψεις.
- **Βελτιστοποιητής:** Ο βελτιστοποιητής Adam επιλέγεται για την αποτελεσματικότητά του και τις προσαρμοστικές δυνατότητες του ρυθμού εκμάθησης. Ο Adam προσαρμόζει δυναμικά τον ρυθμό μάθησης, βελτιστοποιώντας την απόδοση του μοντέλου και σταθεροποιώντας τη διαδικασία εκπαίδευσης.
- **Εποχές και Μέγεθος Παρτίδας:** Το μοντέλο εκπαιδεύεται για 200 εποχές με μέγεθος παρτίδας 8. Μια εποχή είναι ένα πλήρες πέρασμα μέσα από το σύνολο δεδομένων εκπαίδευσης, και το μέγεθος παρτίδας είναι ο αριθμός των δειγμάτων εκπαίδευσης που χρησιμοποιούνται σε μία επανάληψη. [28]

Μετρικές Αξιολόγησης

Η απόδοση του μοντέλου αξιολογείται χρησιμοποιώντας την ακρίβεια, η οποία μετρά την αναλογία των σωστών προβλέψεων. Κατά την εκπαίδευση, η ακρίβεια

παρακολουθείται ως μετρική για την παρακολούθηση της απόδοσης του μοντέλου. Η ακρίβεια υπολογίζεται ως εξής:

Ακρίβεια = Αριθμός Σωστών Προβλέψεων/Συνολικός Αριθμός Προβλέψεων

Παρακολουθώντας την ακρίβεια κατά την εκπαίδευση, μπορούμε να παρατηρήσουμε πόσο καλά το μοντέλο μαθαίνει και προσαρμόζει τις παραμέτρους του.

Sentiment analysis

Η ανάλυση συναισθημάτων είναι ένα σημαντικό πεδίο στη Φυσική Γλώσσας Επεξεργασία (NLP), το οποίο επιδιώκει να κατανοήσει το συναίσθημα ή τη στάση που εκφράζεται σε κείμενο. Δύο δημοφιλείς πλατφόρμες για την υλοποίηση ανάλυσης συναισθημάτων είναι η IBM Watson και η Hugging Face.

IBM Watson:

Η IBM Watson παρέχει ένα εκτεταμένο σύνολο εργαλείων και υπηρεσιών NLP, συμπεριλαμβανομένης της ανάλυσης συναισθημάτων. Η πλατφόρμα χρησιμοποιεί προηγμένες τεχνικές μηχανικής μάθησης και επεξεργασίας γλώσσας για την εξαγωγή και κατηγοριοποίηση συναισθημάτων από κείμενο, προσφέροντας επιλογές προσαρμοσμένες στις ανάγκες των επιχειρήσεων. [29]

Hugging Face:

Για αυτή την πτυχιακή η βιβλιοθήκη που χρησιμοποιήθηκε είναι η hugging face. Η Hugging Face παρέχει ένα αποτελεσματικό περιβάλλον για ανάλυση συναισθήματος μέσω της βιβλιοθήκης Transformers, εκμεταλλευόμενη προηγμένα μοντέλα όπως το BERT, το RoBERTa και άλλα. Εδώ είναι μια γενική επισκόπηση του πώς λειτουργεί η διαδικασία ανάλυσης συναισθήματος με χρήση των εργαλείων της Hugging Face[30]:

Προεπεξεργασία:

- **Διαμερισμός σε Μονάδες (Tokenization):** Το κείμενο εισόδου διαιρείται σε υπομονάδες (tokens) όπως υπολέξεις WordPiece ή SentencePiece που το

μοντέλο μπορεί να κατανοήσει.

- **Συμπλήρωση και Περικοπή:** Βεβαιώνει ότι όλες οι ακολουθίες εισόδου έχουν τον ίδιο μήκος, συνήθως χρησιμοποιώντας συμπλήρωση για μικρότερες ακολουθίες και περικοπή για μεγαλύτερες.

2. Φόρτωση Μοντέλου:

- Η βιβλιοθήκη Transformers της Hugging Face παρέχει πρόσβαση σε μια ποικιλία προ-εκπαιδευμένων μοντέλων κατάλληλων για ανάλυση συναισθήματος. Αυτά τα μοντέλα έχουν εκπαιδευτεί ειδικά σε μεγάλα σύνολα δεδομένων για να επιδοθούν καλά σε συγκεκριμένες εργασίες NLP, συμπεριλαμβανομένης της ανάλυσης συναισθήματος.

3. Ανάλυση:

- Μόλις το μοντέλο φορτωθεί και το κείμενο εισόδου επεξεργαστεί, η διαδικασία ανάλυσης συναισθήματος προβλέπει το συναίσθημα του κειμένου. Αυτή η πρόβλεψη βασίζεται στην έξοδο του μοντέλου, η οποία συνήθως περιλαμβάνει την ανάθεση μιας ετικέτας συναισθήματος (π.χ., θετικό, αρνητικό, ουδέτερο) ή την παροχή βαθμολογίας εμπιστοσύνης για κάθε κατηγορία συναισθήματος.

4. Μεταεπεξεργασία:

- Ανάλογα με τη συγκεκριμένη υλοποίηση και τις απαιτήσεις, η μεταεπεξεργασία μπορεί να περιλαμβάνει τη μετατροπή των εξόδων του μοντέλου σε ανθρώπινες ετικέτες συναισθήματος ή βαθμολογίες.
- Πρόσθετα βήματα μπορεί να περιλαμβάνουν τη χειρισμό των κατωφλίων βεβαιότητας για την κατηγοριοποίηση συναισθημάτων ή την αντιστοίχιση των προβλέψεων του μοντέλου σε συγκεκριμένη κλίμακα συναισθημάτων (π.χ., από 1 έως 5 αστέρια).

Προηγμένες Τεχνικές NLP

Named Entity Recognition (NER)

Η Αναγνώριση Οντοτήτων (NER) είναι μια τεχνική που στοχεύει στον εντοπισμό και την ταξινόμηση ονομάτων οντοτήτων (π.χ., πρόσωπα, οργανισμοί, τοποθεσίες) μέσα σε ένα κείμενο. Αυτή η διαδικασία βοηθά στην κατανόηση της σημασίας και της σχέσης των οντοτήτων σε ένα κείμενο.

Τεχνολογίες και εργαλεία NER:

- **SpaCy:** Ένα δημοφιλές λογισμικό NLP που περιλαμβάνει προ-εκπαιδευμένα μοντέλα για την αναγνώριση οντοτήτων.
- **Hugging Face Transformers:** Προσφέρει μοντέλα όπως το BERT που μπορούν να χρησιμοποιηθούν για την αναγνώριση οντοτήτων.

Text Summarization

Η Συνοπτική Αναπαράσταση Κειμένου (Text Summarization) είναι η διαδικασία της αυτόματης δημιουργίας σύντομων, περιεκτικών και ουσιαστικών περιλήψεων από μεγαλύτερα κείμενα. Υπάρχουν δύο κύριες προσεγγίσεις:

1. **Extractive Summarization:** Επιλέγει και εξάγει τις πιο σημαντικές προτάσεις από το αρχικό κείμενο.
2. **Abstractive Summarization:** Δημιουργεί νέες προτάσεις που συνοψίζουν το περιεχόμενο του αρχικού κειμένου, χρησιμοποιώντας μοντέλα βαθιάς μάθησης.

Τεχνολογίες και εργαλεία για Text Summarization:

- **T5 (Text-To-Text Transfer Transformer):** Ένα ισχυρό μοντέλο από την Google που μετατρέπει διάφορες εργασίες NLP σε προβλήματα μετατροπής

κειμένου.

- **GPT (Generative Pre-trained Transformer):** Χρησιμοποιείται για τη δημιουργία περιλήψεων με βάση τη βαθιά κατανόηση του περιεχομένου.

2.2.1.i Large Language Models

Τα Μεγάλα Μοντέλα Γλώσσας (LLMs)

Τα Μεγάλα Μοντέλα Γλώσσας (Large Language Models, LLMs) είναι προηγμένα μοντέλα τεχνητής νοημοσύνης σχεδιασμένα να κατανοούν και να παράγουν κείμενο που μοιάζει με ανθρώπινο. Χρησιμοποιούν τεχνικές βαθιάς μάθησης, ιδιαίτερα νευρωνικά δίκτυα με αρχιτεκτονική μετασχηματιστών (transformers), για να επεξεργάζονται και να αναλύουν τεράστιες ποσότητες κειμένου.

Αρχιτεκτονική Μετασχηματιστών

Η αρχιτεκτονική των μετασχηματιστών (Transformers) εισήχθη από τους Vaswani et al. (2017) και έχει αλλάξει ριζικά την επεξεργασία φυσικής γλώσσας. Σε αντίθεση με τα παραδοσιακά RNNs (Recurrent Neural Networks), οι μετασχηματιστές επιτρέπουν την παράλληλη επεξεργασία δεδομένων, βελτιώνοντας έτσι την αποδοτικότητα και την ακρίβεια των μοντέλων.

Εκπαίδευση σε Μεγάλα Δεδομένα

Τα LLMs εκπαιδεύονται σε εκτεταμένα δεδομένα από διάφορες πηγές κειμένου, όπως βιβλία, άρθρα, ιστοσελίδες, και κοινωνικά μέσα. Με αυτόν τον τρόπο, μαθαίνουν τις λεπτομέρειες της γλώσσας, το πλαίσιο και τη σημασία. Ένα από τα πιο γνωστά LLMs είναι το GPT-3 (Generative Pre-trained Transformer 3), το οποίο έχει 175 δισεκατομμύρια παραμέτρους.

Δυνατότητες και Εφαρμογές

Οι δυνατότητες των LLMs περιλαμβάνουν:

- **Δημιουργία Περιεχομένου:** Μπορούν να δημιουργούν συνεκτικό και σχετικό κείμενο για διάφορες εφαρμογές, όπως άρθρα, αναφορές, και δημιουργική γραφή.
- **Συνομιλητικοί Πράκτορες:** Χρησιμοποιούνται σε chatbots και εικονικούς βοηθούς για να απαντούν σε ερωτήσεις και να παρέχουν υποστήριξη σε πραγματικό χρόνο.
- **Αυτόματη Μετάφραση Γλώσσας:** Βελτιώνουν την ακρίβεια και τη φυσικότητα της μετάφρασης κειμένου μεταξύ διαφορετικών γλωσσών.
- **Ανάλυση Δεδομένων:** Βοηθούν στην ανάλυση μεγάλων συνόλων δεδομένων και στην εξαγωγή χρήσιμων πληροφοριών.
- **Εκπαιδευτικά Εργαλεία:** Χρησιμοποιούνται σε εφαρμογές εκμάθησης και εκπαίδευσης για τη δημιουργία περιεχομένου και τη διαδραστική διδασκαλία .

Προκλήσεις και Περιορισμοί

Ωστόσο, η εκπαίδευση και η χρήση των LLMs συνοδεύονται από σημαντικές προκλήσεις:

- **Υπολογιστικοί Πόροι:** Απαιτούν τεράστιους υπολογιστικούς πόρους για την εκπαίδευσή τους, που μπορεί να είναι δαπανηροί και περιβαλλοντικά επιβαρυντικοί .
- **Προκαταλήψεις:** Μπορεί να κληρονομήσουν και να αναπαράγουν προκαταλήψεις από τα δεδομένα εκπαίδευσης, οδηγώντας σε μεροληπτικές ή άδικες αποφάσεις .
- **Διαφάνεια και Εξηγησιμότητα:** Η κατανόηση της διαδικασίας λήψης αποφάσεων τους είναι σύνθετη, εγείροντας ζητήματα διαφάνειας και εξηγησιμότητας .
- **Ηθική και Ιδιωτικότητα:** Εγείρονται θέματα ηθικής και ιδιωτικότητας δεδομένων, ιδιαίτερα όταν χρησιμοποιούνται σε ευαίσθητες εφαρμογές .

2.2.2 *Natural Language Understanding*

Η Κατανόηση Φυσικής Γλώσσας (Natural Language Understanding - NLU) είναι ένα κρίσιμο συστατικό της επεξεργασίας φυσικής γλώσσας (Natural Language Processing - NLP), το οποίο επικεντρώνεται στη δυνατότητα των μηχανών να κατανοούν και να ερμηνεύουν την ανθρώπινη γλώσσα με τρόπο που είναι χρήσιμος και επεξεργάσιμος από την μηχανή. Η NLU περιλαμβάνει μια σειρά από εργασίες που στοχεύουν στην εξαγωγή σημασιολογικής σημασίας από κείμενο ή ομιλία, διευκολύνοντας εφαρμογές όπως οι εικονικοί βοηθοί, η ανάλυση συναισθημάτων και τα αυτοματοποιημένα συστήματα υποστήριξης πελατών.

Κύριες Εργασίες της NLU:

1. Διαχωρισμός και Τμηματοποίηση:

- **Διαχωρισμός (Tokenization):** Ανάλυση κειμένου σε μικρότερες μονάδες όπως λέξεις ή υπολέξεις, συμπεριλαμβανομένων σημείων στίξης και ειδικών χαρακτήρων.
- **Τμηματοποίηση (Segmentation):** Διαίρεση μακρών κειμένων ή συνεχούς ομιλίας σε μικρότερα, αναλύσιμα τμήματα.

2. Επισήμανση Μέρους του Λόγου (Part-of-Speech - POS):

- Αναγνώριση των γραμματικών μερών του λόγου (π.χ., ουσιαστικό, ρήμα, επίθετο) κάθε λέξης σε μια πρόταση. Αυτό βοηθά στην κατανόηση της συντακτικής δομής του κειμένου.

3. Αναγνώριση Ονομαστικών Οντοτήτων (Named Entity Recognition - NER):

- Αναγνώριση και κατηγοριοποίηση ονομαστικών οντοτήτων (όπως

ονόματα ατόμων, οργανισμών, τοποθεσιών) μέσα σε ένα κείμενο. Αυτό είναι κρίσιμο για την εξαγωγή χρήσιμων πληροφοριών από μη δομημένα δεδομένα.

4. Συντακτική Ανάλυση (Syntactic Parsing):

- Ανάλυση της γραμματικής δομής των προτάσεων για τον προσδιορισμό του τρόπου με τον οποίο οι λέξεις σχετίζονται μεταξύ τους, βοηθώντας στην κατανόηση της σημασίας της πρότασης πέρα από τις μεμονωμένες λέξεις.

5. Επισημάνση Σημασιολογικών Ρόλων (Semantic Role Labeling - SRL):

- Αναγνώριση των σχέσεων μεταξύ των λέξεων σε μια πρόταση και των αντίστοιχων ρόλων τους (π.χ., πράκτορας, ασθενής, εργαλείο) στην πράξη ή το γεγονός που περιγράφεται. Αυτό βοηθά στην κατανόηση της σημασίας των προτάσεων ως προς πράξεις και συμμετέχοντες.

6. Επίλυση Συνδετικής Αναφοράς (Coreference Resolution):

- Επίλυση των αναφορών στο κείμενο που αναφέρονται στην ίδια οντότητα. Για παράδειγμα, κατανόηση ότι "αυτός" και "Γιάννης" αναφέρονται στον ίδιο άνθρωπο σε συγκεκριμένο πλαίσιο.

7. Ανάλυση Συναισθήματος (Sentiment Analysis):

- Πρόβλεψη του συναισθήματος ή της γνώμης που εκφράζεται σε ένα κείμενο, η οποία είναι κρίσιμη για εφαρμογές όπως η παρακολούθηση κοινωνικών μέσων και η ανάλυση των ανατροφοδοτήσεων πελατών.

Terminology: NLU vs. NLP vs. ASR

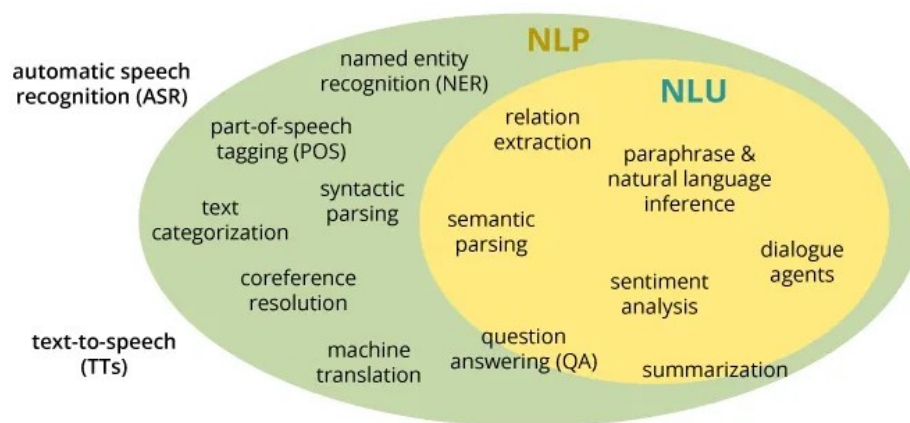


Figure 6. Terminology: NLU vs NLP vs ASR [\[31\]](#)

2.2.3 Διάσημα frameworks για chatbots

2.2.3.i Rasa

Το Rasa είναι ένα από τα πιο γνωστά πλαίσια ανοιχτού κώδικα για τη δημιουργία chatbots, το οποίο παρέχει μεγάλη ευελιξία και προσαρμοστικότητα. Ένα από τα κύρια πλεονεκτήματα του Rasa είναι ότι επιτρέπει την τοπική ανάπτυξη, δίνοντας στους προγραμματιστές πλήρη έλεγχο των δεδομένων και των διαδικασιών. Αυτό είναι ιδιαίτερα σημαντικό για οργανισμούς που επιθυμούν να διατηρήσουν τα δεδομένα τους ασφαλή και ιδιωτικά. Το Rasa υποστηρίζει την ενσωμάτωση με διάφορες εξωτερικές υπηρεσίες και βάσεις δεδομένων, διευκολύνοντας την προσαρμογή των bots σύμφωνα με τις συγκεκριμένες ανάγκες των χρηστών. [\[32\]](#)

Ωστόσο, παρά τα πολλά του πλεονεκτήματα, το Rasa μπορεί να είναι απαιτητικό για χρήστες χωρίς εμπειρία προγραμματισμού, καθώς η αρχική ρύθμιση και συντήρηση του απαιτεί τεχνικές γνώσεις και πόρους. Επίσης, η έλλειψη ενσωματωμένων εργαλείων διαχείρισης μπορεί να κάνει τη διαχείριση των projects πιο περίπλοκη.

Θετικά:

- Πλήρης έλεγχος των δεδομένων
- Εξαιρετική ευελιξία και προσαρμοστικότητα
- Ανοιχτός κώδικας

Αρνητικά:

- Απαιτεί τεχνικές γνώσεις για την εγκατάσταση και συντήρηση
- Περιορισμένα ενσωματωμένα εργαλεία διαχείρισης

2.2.3.ii Dialogflow

Το Dialogflow, που ανήκει στην Google, είναι ένα ισχυρό εργαλείο για τη δημιουργία chatbots χρησιμοποιώντας φυσική επεξεργασία γλώσσας (NLP). Ένα από τα μεγαλύτερα πλεονεκτήματά του είναι η ευκολία χρήσης, με ένα φιλικό προς τον χρήστη περιβάλλον ανάπτυξης που διευκολύνει ακόμη και αρχάριους χρήστες να δημιουργήσουν ισχυρά chatbots. Το Dialogflow είναι ενσωματωμένο με τον Google Assistant, καθώς και με άλλες πλατφόρμες όπως το Facebook Messenger και το Slack, επιτρέποντας στους προγραμματιστές να δημιουργούν bots που λειτουργούν σε πολλαπλές πλατφόρμες ταυτόχρονα. []

Οι δυνατότητες ανάλυσης συναισθημάτων και επεξεργασίας πολλαπλών γλωσσών καθιστούν το Dialogflow μια εξαιρετική επιλογή για παγκόσμιες εφαρμογές. Ωστόσο, καθώς αυξάνονται οι απαιτήσεις του έργου, η πολυπλοκότητα της διαχείρισης των intent και των entities μπορεί να αυξηθεί, απαιτώντας μεγαλύτερη προσοχή και γνώση.

Θετικά:

- Εύκολο στη χρήση με φιλικό προς τον χρήστη περιβάλλον ανάπτυξης
- Άριστη ενσωμάτωση με πλατφόρμες Google και άλλες υπηρεσίες
- Δυνατότητες ανάλυσης συναισθημάτων και υποστήριξη πολλαπλών γλωσσών

Αρνητικά:

- Η διαχείριση των intent και των entities μπορεί να γίνει πολύπλοκη σε μεγάλα projects
- Μπορεί να υπάρχουν περιορισμοί στην προσαρμογή

2.2.3.iii Landbot AI

Το Landbot AI είναι ένα ευέλικτο πλαίσιο για τη δημιουργία chatbots με έμφαση στη σχεδίαση και την εμπειρία χρήστη. Το Landbot AI προσφέρει ένα οπτικό περιβάλλον δημιουργίας που επιτρέπει στους χρήστες να δημιουργούν ροές συνομιλιών χρησιμοποιώντας drag-and-drop λειτουργίες, κάνοντάς το ιδανικό για χρήστες χωρίς προγραμματιστικές γνώσεις. Η πλατφόρμα επιτρέπει την εύκολη ενσωμάτωση με διάφορες υπηρεσίες, όπως CRM και εργαλεία marketing, παρέχοντας έναν εύκολο τρόπο για την αυτοματοποίηση της επικοινωνίας με τους πελάτες.

Παρά το γεγονός ότι το Landbot AI είναι εξαιρετικά προσβάσιμο και εύκολο στη χρήση, μπορεί να μην διαθέτει την ίδια ισχύ και προσαρμοστικότητα με τα πιο τεχνικά πλαίσια όπως το Rasa, ιδιαίτερα για πολύπλοκες εφαρμογές που απαιτούν εκτεταμένη επεξεργασία γλώσσας. [\[34\]](#)

Θετικά:

- Εύκολο στη χρήση με οπτικό περιβάλλον δημιουργίας
- Ιδανικό για χρήστες χωρίς προγραμματιστικές γνώσεις
- Εύκολη ενσωμάτωση με διάφορες υπηρεσίες

Αρνητικά:

- Περιορισμένη ισχύς και προσαρμοστικότητα σε σύγκριση με πιο τεχνικά πλαίσια

- Μπορεί να μην είναι κατάλληλο για πολύπλοκες εφαρμογές

2.2.3.iv TensorFlow

Το TensorFlow είναι ένα framework ανοιχτού κώδικα για μηχανική μάθηση που αναπτύχθηκε από την Google. Χρησιμοποιείται ευρέως για την ανάπτυξη και την εκπαίδευση μοντέλων μηχανικής μάθησης και βαθιάς μάθησης, συμπεριλαμβανομένων των chatbots. Ένα από τα μεγάλα πλεονεκτήματα του TensorFlow είναι η ισχύς και η ευελιξία του, επιτρέποντας στους προγραμματιστές να δημιουργήσουν πολύπλοκα μοντέλα που μπορούν να επεξεργαστούν φυσική γλώσσα με υψηλή ακρίβεια. [35]

Η ενσωμάτωση με άλλες βιβλιοθήκες της Google, όπως το TensorFlow Extended (TFX) και το TensorFlow Lite, επιτρέπει την ανάπτυξη και τη διάθεση μοντέλων σε παραγωγικό περιβάλλον και κινητές συσκευές αντίστοιχα. Ωστόσο, το TensorFlow μπορεί να είναι πολύπλοκο για αρχάριους και απαιτεί καλή γνώση της μηχανικής μάθησης και του προγραμματισμού για την αποτελεσματική χρήση του. Παρόλα αυτά, η ισχύς και η ευελιξία του το καθιστούν μια εξαιρετική επιλογή για προχωρημένες εφαρμογές που απαιτούν υψηλή απόδοση και προσαρμοστικότητα.

Θετικά:

- Ισχυρό και ευέλικτο πλαίσιο για μηχανική μάθηση και βαθιά μάθηση
- Υψηλή ακρίβεια στην επεξεργασία φυσικής γλώσσας
- Ενσωμάτωση με άλλες βιβλιοθήκες της Google

Αρνητικά:

- Πολύπλοκο για αρχάριους χρήστες
- Απαιτεί καλή γνώση της μηχανικής μάθησης και του προγραμματισμού

3. ΑΝΑΛΥΣΗ ΚΩΔΙΚΑ

Το τρίτο κομμάτι αυτού της πτυχιακής αναφαίρεται στην υλοποίηση του chatbot και περιέχει λεπτομερή ανάλυση των βιβλιοθηκών, των διαφόρων εντολών και το γενικότερο pipeline του κώδικα. Επίσης γίνονται αναφορές σε προβλήματα που εμφανίστηκαν και μελλοντικές βελτιώσεις.

Περίληψη διαδικασίας:

Είσοδος Χρήστη: Το chatbot λαμβάνει ένα μήνυμα από τον χρήστη. Αυτή η είσοδος μπορεί να είναι:

- Ένας σύνδεσμος Spotify
- Ένα όνομα καλλιτέχνη
- Ένα όνομα τραγουδιού
- Ένα είδος μουσικής
- Ένα συναίσθημα ή διάθεση

- **Κατηγοριοποίηση Εισόδου:**

- **Χειρισμός Συνδέσμων:** Εάν η είσοδος είναι ένας σύνδεσμος Spotify, το chatbot εξάγει τον τύπο (τραγούδι, καλλιτέχνης, λίστα αναπαραγωγής, άλμπουμ) και το ID.
- **Χειρισμός Κειμένου:** Εάν η είσοδος είναι κείμενο, το chatbot καθορίζει εάν αναφέρεται σε καλλιτέχνη, τραγούδι, είδος μουσικής ή συναίσθημα.

- **Ανάκτηση Δεδομένων:**

- **Για Συνδέσμους:** Το chatbot χρησιμοποιεί το API του Spotify για να

ανακτήσει σχετικές λεπτομέρειες βάσει του τύπου και του ID που εξήχθη.

- **Για Κείμενο:**

- Εάν η είσοδος είναι καλλιτέχνης ή τραγούδι, το chatbot χρησιμοποιεί το API του Spotify για να αναζητήσει αντίστοιχα στοιχεία.
- Εάν η είσοδος είναι είδος μουσικής ή συναίσθημα, το chatbot χαρτογραφεί την είσοδο σε μια λίστα αντίστοιχων ειδών ή διαθέσεων και ανακτά σχετικά τραγούδια/καλλιτέχνες από το Spotify.

- **Υπολογισμός Ομοιότητας:**

- Βάσει των ανακτηθέντων δεδομένων, το chatbot χρησιμοποιεί το API συστάσεων του Spotify για να βρει παρόμοια τραγούδια. Τα κριτήρια ομοιότητας εξαρτώνται από τον τύπο εισόδου:
 - **Τραγούδι/Καλλιτέχνης:** Το chatbot ανακτά παρόμοια τραγούδια βάσει των χαρακτηριστικών του τραγουδιού/καλλιτέχνη εισόδου.
 - **Είδος Μουσικής/Συναίσθημα:** Το chatbot χρησιμοποιεί τις συστάσεις του Spotify για τραγούδια που ταιριάζουν με το είδος ή τη διάθεση εισόδου.

- **Διαμόρφωση Απάντησης:**

- Το chatbot εξάγει σχετικές λεπτομέρειες (όνομα τραγουδιού, όνομα καλλιτέχνη, σύνδεσμο Spotify, προεπισκόπηση URL) για κάθε προτεινόμενο τραγούδι.
- Το chatbot διαμορφώνει αυτές τις λεπτομέρειες σε ένα φιλικό προς τον χρήστη μήνυμα.

- **Παράδοση Απάντησης:**

- Το chatbot στέλνει τη διαμορφωμένη απάντηση πίσω στον χρήστη.

Παράδειγμα

:

Σενάριο 1: Ο Χρήστης Παρέχει Έναν Σύνδεσμο Spotify

- **Είσοδος:** <https://open.spotify.com/track/4uLU6hMCjMI75M1A2tKUQC>
- **Κατηγοριοποίηση:** Το chatbot αναγνωρίζει την είσοδο ως σύνδεσμο τραγουδιού Spotify.
- **Ανάκτηση Δεδομένων:** Το chatbot χρησιμοποιεί το API του Spotify για να λάβει λεπτομέρειες του τραγουδιού με ID 4uLU6hMCjMI75M1A2tKUQC.
- **Υπολογισμός Ομοιότητας:** Το chatbot ανακτά παρόμοια τραγούδια χρησιμοποιώντας το API συστάσεων του Spotify.
- **Διαμόρφωση Απάντησης:** Το chatbot διαμορφώνει λεπτομέρειες των προτεινόμενων τραγουδιών.
- **Παράδοση Απάντησης:** Το chatbot στέλνει την απάντηση στον χρήστη.

Σενάριο 2: Ο Χρήστης Αναφέρει Έναν Καλλιτέχνη

- **Είσοδος:** "Μου αρέσει το συγκροτιμα Coldplay"
- **Κατηγοριοποίηση:** Το chatbot αναγνωρίζει το "Coldplay" ως καλλιτέχνη.
- **Ανάκτηση Δεδομένων:** Το chatbot χρησιμοποιεί το API του Spotify για να αναζητήσει τον Coldplay.
- **Υπολογισμός Ομοιότητας:** Το chatbot ανακτά τραγούδια παρόμοια με τη μουσική του Coldplay.
- **Διαμόρφωση Απάντησης:** Το chatbot διαμορφώνει λεπτομέρειες των προτεινόμενων τραγουδιών.
- **Παράδοση Απάντησης:** Το chatbot στέλνει την απάντηση στον χρήστη.

Σενάριο 3: Ο Χρήστης Περιγράφει Ένα Συναίσθημα

- **Είσοδος:** "Νιώθω χαρούμενος"
- **Κατηγοριοποίηση:** Το chatbot αναγνωρίζει το "χαρούμενος" ως συναίσθημα.

- **Ανάκτηση Δεδομένων:** Το chatbot χαρτογραφεί το "χαρούμενος" σε κατάλληλα είδη/διαθέσεις (π.χ., "pop", "upbeat") και ανακτά σχετικά τραγούδια.
- **Υπολογισμός Ομοιότητας:** Το chatbot ανακτά τραγούδια που ταιριάζουν με τα χαρτογραφημένα είδη/διαθέσεις.
- **Διαμόρφωση Απάντησης:** Το chatbot διαμορφώνει λεπτομέρειες των προτεινόμενων τραγουδιών.
- **Παράδοση Απάντησης:** Το chatbot στέλνει την απάντηση στον χρήστη.

3.1 Imports και βιβλιοθήκες

Ο κώδικας χρησιμοποιεί τεχνολογίες νέφους και τρέχει όλα τα απαραίτητα κομμάτια στο cloud με την χρήση του google colab. Αν και ο κώδικας μπορεί να τρέξει και locally όλα τα παραδείγματα που θα εμφανιστούν είναι μέσω colab. Έτσι πρώτα από όλα πρέπει να γίνει ένα βασικό import των αρχείων από το google drive. Το βασικότερο αρχείο για τώρα είναι το intents.json. Ως intents σε ένα τέτοιο chatbot έχουμε ορίσει κάποιες βασικές απαντήσεις που θέλουμε να απαντάει με συγκεκριμένο τρόπο το chatbot. Για παράδειγμα άμα ο χρήστης θέλει να ρωτήσει “Πως σε λένε” αυτό είναι κάτι που δεν μπορεί να βρει εκτός αν το έχουμε ορίσει εμείς.

```
},
{
  "tag": "name1",
  "patterns": ["what's your name?", "who are you?"],
  "responses": ["I'm Melody, your musical chatbot!", "Melody at your service! How can I assist you today?"]
},
{
  "tag": "jokes",
  "patterns": ["Tell me a joke", "I need a laugh", "Make me laugh", "Got any jokes?"],
  "responses": [
    "Why did the music teacher go to jail? Because she got caught with too many sharp objects!",
    "How do you fix a broken tuba? With a tuba glue!",
    "What type of music are balloons afraid of? Pop music!"
  ]
}
```

Για να το κάνουμε αυτό πρέπει να κάνουμε import το google drive.

```
[ ] from google.colab import drive
    drive.mount('/content/drive')
```

Εκτός από το intents.json έχει και άλλα αρχεία όπως το punkt και το wordnet τα οποία θα τα αναλύσουμε πιο μετά.

Έπειτα κάθε φορά που ξεκινάει το colab πρέπει να γίνεται ξανά η εγκατάσταση κάποιων βιβλιοθηκών όπως το nltk, requests και άλλα

```
!pip install nltk numpy tensorflow requests spotipy
```

```
Collecting redis>=3.5.3 (from spotipy)
  Downloading redis-5.0.4-py3-none-any.whl (251 kB)
  252.0/252.0 kB 2.1 MB/s eta 0:00:00
```

Έτσι αφού τειρούνται όλα αυτά ξεκινάει το import όλων των βιβλιοθηκών:

Επεξήγηση βιβλιοθηκών:

```
import numpy as np
import tensorflow as tf
import random
import requests
import json
from spotipy.oauth2 import SpotifyClientCredentials
import spotipy
import nltk
from nltk.stem import WordNetLemmatizer
from nltk.tokenize import word_tokenize
from nltk.util import bigrams as nltk_bigrams
#from textblob import TextBlob
from transformers import pipeline
```

tensorflow (import tensorflow ως tf):

Το tensorflow είναι ένα open-source machine learning framework αναπτυγμένο από την Google. Χρησιμοποιείται ευρέως για την κατασκευή και εκπαίδευση διάφορων τύπων μοντέλων βαθιάς μάθησης, συμπεριλαμβανομένων των νευρωνικών δικτύων.

random:

Το module random παρέχει συναρτήσεις για την παραγωγή τυχαίων αριθμών, την ανακάτεμση ακολουθιών και την επιλογή τυχαίων στοιχείων.

requests:

Το requests είναι μια δημοφιλής βιβλιοθήκη HTTP για την αποστολή HTTP αιτημάτων στην Python. Απλοποιεί την αποστολή αιτημάτων HTTP και την αντιμετώπιση των απαντήσεων.

json:

Το module json παρέχει συναρτήσεις για την εργασία με δεδομένα JSON (JavaScript Object Notation), ένα ελαφρύ μορφότυπο ανταλλαγής δεδομένων. Στην δικιά μας περίπτωση χρησιμοποιείται για το άνοιγμα των intents.

Spotipy:

Το Spotipy είναι μια βιβλιοθήκη Python για την πρόσβαση στο Spotify Web API. Παρέχει μεθόδους για την αναζήτηση μουσικής, την ανάκτηση συστάσεων και τη διαχείριση λιστών αναπαραγωγής του χρήστη.

spotify.oauth2.SpotifyClientCredentials:

Το SpotifyClientCredentials είναι μέρος της βιβλιοθήκης Spotipy, η οποία είναι ένας πελάτης Python για το Spotify Web API. Διαχειρίζεται τη ροή εξουσιοδότησης πελάτη για την αυθεντικοποίηση αιτημάτων προς το Spotify API.

nltk (import nltk):

Το NLTK (Natural Language Toolkit) είναι μια βιβλιοθήκη για επεξεργασία φυσικής γλώσσας (NLP) στην Python. Παρέχει μονάδες για το tokenization, τη στελέχωση, το tagging, την ανάλυση σύνταξης και άλλα.

nltk.stem.WordNetLemmatizer:

Ο WordNetLemmatizer χρησιμοποιείται για την ελαχιστοποίηση των λέξεων στο NLTK. Η ελαχιστοποίηση μειώνει τις λέξεις στην βάση τους ή στο λεξικό τους σχήμα (λήμματα).

nltk.tokenize.word_tokenize:

Το word_tokenize είναι ένα εργαλείο διάσπασης που παρέχεται από το NLTK, χρησιμοποιείται για τη διαίρεση κειμένου σε διακριτά τμήματα (λέξεις και σημεία στίξης).

nltk.util.bigrams as nltk_bigrams:

Το nltk_bigrams είναι ένα ψευδώνυμο για τη δημιουργία μεγάλων διαστάσεων (ακολουθίες δύο διαδοχικών στοιχείων) από μια λίστα τμημάτων.

transformers.pipeline:

Η συνάρτηση pipeline από τη βιβλιοθήκη Hugging Face Transformers διαμορφώνει ένα αίματος επεξεργασίας φυσικής γλώσσας.

3.2 Natural Language Processing

Το μοντέλο που χρησιμοποιήθηκε για αυτό το chatbot δεν είναι πολύ περίπλοκο μιας και δεν ήταν απαραίτητο για την τωρινή χρήση του. Μπορεί να κατανοήσει το input του χρήστη, να ταξινομήσει τις προθέσεις του και να παρέχει κατάλληλες απαντήσεις.

Κατηγοριοποίηση και λημματοποίηση

Κατηγοριοποίηση είναι η διαδικασία διαχωρισμού του κειμένου σε μεμονωμένες λέξεις ή σύμβολα. Αυτό βοηθά στην κατανόηση της δομής και της σημασίας των προτάσεων. Η **Λημματοποίηση** είναι η διαδικασία μείωσης των λέξεων στη βασική τους μορφή. Για παράδειγμα, η λέξη "running" γίνεται "run".

Στον κώδικα αυτά τα δύο πραγματοποιούνται χρησιμοποιώντας το NLTK (Natural Language Toolkit)

```
# Initialize lemmatizer and sentiment analyzer
lemmatizer = WordNetLemmatizer()
```

```
# Tokenizes and lemmatizes a sentence, and generates bigrams
def clean_up_sentence(sentence):
    sentence_words = word_tokenize(sentence) # Tokenize the sentence
    sentence_words = [lemmatizer.lemmatize(word.lower()) for word in sentence_words] # Lemmatize each word
    # Generate bigrams from the list of words
    sentence_bigrams = [' '.join(bigram) for bigram in nltk_bigrams(sentence_words)]
    return sentence_words + sentence_bigrams # Return a list of words and bigrams
```

Δίγραμματα

Τα **δίγραμματα** είναι ζεύγη διαδοχικών λέξεων σε μια πρόταση. Καταγράφουν το πλαίσιο των ζεύγων λέξεων, που μπορεί να είναι σημαντικό για την κατανόηση

φράσεων και ιδιωμάτων. Για παράδειγμα, στην πρόταση "I love music," το δίγραμμα είναι "I love" και "love music." Αυτό μας επιτρέπει στο να πιάσουμε πιο περίεργες περιπτώσεις ή στο μέλλον να έχουμε πιο περίεργα genres μουσικής (π.χ punk rock).

```
# Generate bigrams from the list of words
sentence_bigrams = [' '.join(bigram) for bigram in nltk_bigrams(sentence_words)]
return sentence_words + sentence_bigrams # Return a list of words and bigrams
```

Bag of words

Το **Bag of Words** μοντέλο μετατρέπει το κείμενο σε σύνολο διανυσμάτων. Κάθε διάνυσμα αντιπροσωπεύει την παρουσία (1) ή απουσία (0) λέξεων και διγράμμάτων από ένα προκαθορισμένο λεξιλόγιο. Αυτό είναι κρίσιμο για τη μετατροπή του κειμένου σε μορφή κατάλληλη για μοντέλα μηχανικής μάθησης.

```
# Converts a sentence into a bag of words representation
def bow(sentence, words):
    sentence_words_bigrams = clean_up_sentence(sentence) # Get both words and bigrams
    bag = [1 if w in sentence_words_bigrams else 0 for w in words] # Create bag of words array for both
    return np.array(bag)
```

Ταξινόμηση Προθέσεων με Νευρωνικά Δίκτυα

Το chatbot χρησιμοποιεί ένα **νευρωνικό δίκτυο** για την ταξινόμηση των προθέσεων των χρηστών. Συγκεκριμένα, χρησιμοποιείται ένα **Sequential μοντέλο** από τη βιβλιοθήκη Keras του TensorFlow. Το Sequential μοντέλο επιτρέπει τη στοίχιση επιπέδων σε γραμμική μορφή, η οποία είναι κατάλληλη για πολλούς τύπους νευρωνικών δικτύων.

Το μοντέλο αποτελείται από:

- Ένα επίπεδο εισόδου που αντιστοιχεί στο μέγεθος των BoW διανυσμάτων.
- Δύο κρυφά επίπεδα με ReLU (Rectified Linear Unit) ενεργοποίηση και

dropout για κανονικοποίηση.

- Ένα εξόδου επίπεδο με softmax ενεργοποίηση για την ταξινόμηση των προθέσεων.

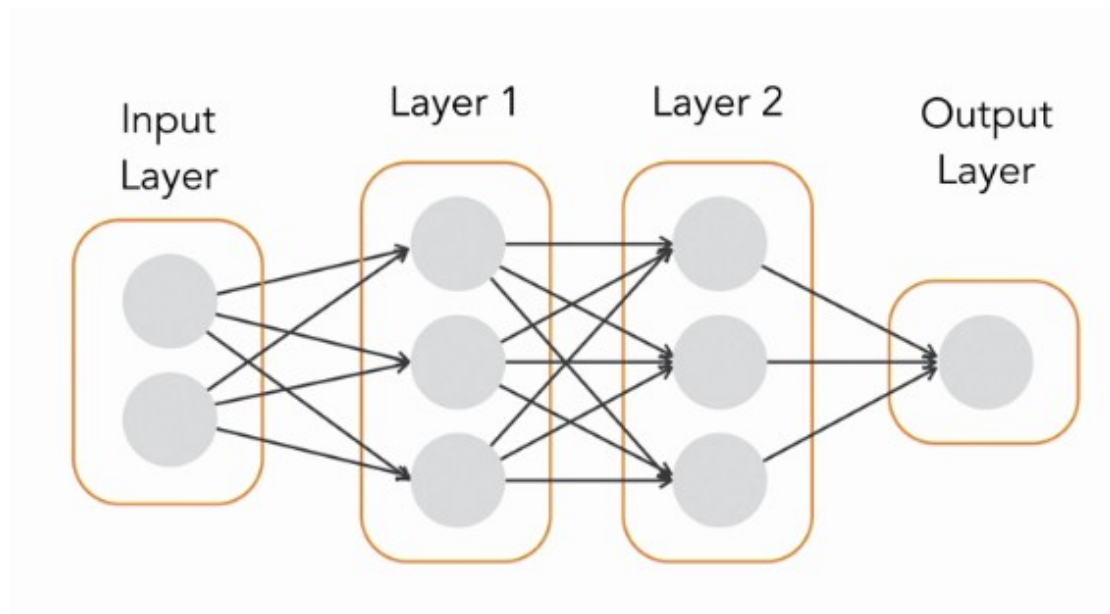


Figure 7. Keras Sequential model [36]

```
# Define a neural network with an input layer, two hidden layers (with dropout for regularization),
# and an output layer with softmax activation.
# Compiles the model with categorical_crossentropy loss and the Adam optimizer
# Finally, train the model on the preprocessed data and save it
model = tf.keras.Sequential([
    tf.keras.layers.InputLayer(input_shape=(len(train_x[0]),)),
    tf.keras.layers.Dense(128, activation='relu'),
    tf.keras.layers.Dropout(0.5),
    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dropout(0.5),
    tf.keras.layers.Dense(len(train_y[0]), activation='softmax')
])

model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
model.fit(train_x, train_y, epochs=200, batch_size=8)
model.save('chatbot_model')
```

Η εκπαίδευση περιλαμβάνει τη σίτιση του μοντέλου με προεπεξεργασμένα δεδομένα (BoW διανύσματα και αντίστοιχες ετικέτες προθέσεων) και την προσαρμογή των βαρών για την ελαχιστοποίηση της συνάρτησης απώλειας. Τα δεδομένα εκπαίδευσης ανακατεύονται για να διασφαλιστεί ότι το μοντέλο γενικεύει καλά. Χρησιμοποιείται η συνάρτηση απώλειας **categorical_crossentropy** και ο βελτιστοποιητής **adam** για την εκπαίδευση.

```
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
model.fit(train_x, train_y, epochs=200, batch_size=8)
model.save('chatbot_model')
```

Ολοκλήρωση

- 1. Είσοδος Χρήστη:** Ο χρήστης παρέχει μια εισαγωγική πρόταση.
- 2. Κατηγοριοποίηση και Λημματοποίηση:** Η είσοδος κατηγοριοποιείται και λημματοποιείται.
- 3. Bag of Words:** Οι επεξεργασμένες λέξεις και δίγραμματα μετατρέπονται σε BoW διάνυσμα.
- 4. Ταξινόμηση Προθέσεων:** Το BoW διάνυσμα τροφοδοτείται στο νευρωνικό δίκτυο για την ταξινόμηση της πρόθεσης.
- 5. Ανάλυση Συναισθήματος:** Αναλύεται το συναίσθημα της εισαγωγής για τον προσδιορισμό της συναισθηματικής διάθεσης.
- 6. Δημιουργία Απάντησης:** Βασισμένο στην πρόθεση και τη διάθεση, δημιουργούνται κατάλληλες απαντήσεις.
- 7. Διαχείριση Συνεδρίας:** Το πλαίσιο διατηρείται σε πολλαπλές αλληλεπιδράσεις χρησιμοποιώντας συνεδρίες.

Sentiment analysis

3.2.1 *Τι είναι ένα Pipeline;*

Στο πλαίσιο της επεξεργασίας φυσικής γλώσσας (NLP), ένα pipeline αναφέρεται σε μια σειρά συνιστωσών επεξεργασίας δεδομένων που συνδέονται αλυσιδωτά μεταξύ τους. Κάθε συνιστώσα στον πίνακα επεξεργάζεται κάποιο κομμάτι των εισερχόμενων δεδομένων και περνά την έξοδο της στην επόμενη συνιστώσα. Οι πίνακες είναι συνηθισμένοι στις εργασίες NLP για την οργάνωση και τη ροή της προεπεξεργασίας δεδομένων, της εξαγωγής χαρακτηριστικών και της πρόβλεψης μοντέλων.

3.2.2 *Η Βιβλιοθήκη Transformers από την Hugging Face*

Η βιβλιοθήκη transformers από την Hugging Face παρέχει μια εύκολη διεπαφή για την εργασία με προηγμένα μοντέλα μετασχηματιστών στην NLP. Οι μετασχηματιστές είναι μοντέλα νευρωνικών δικτύων που ξεχωρίζουν σε εργασίες όπως η κατανόηση γλώσσας και η δημιουργία. Χρησιμοποιούνται ευρέως για εργασίες όπως η ανάλυση συναισθήματος, η ταξινόμηση κειμένου και η μετάφραση μηχανής.

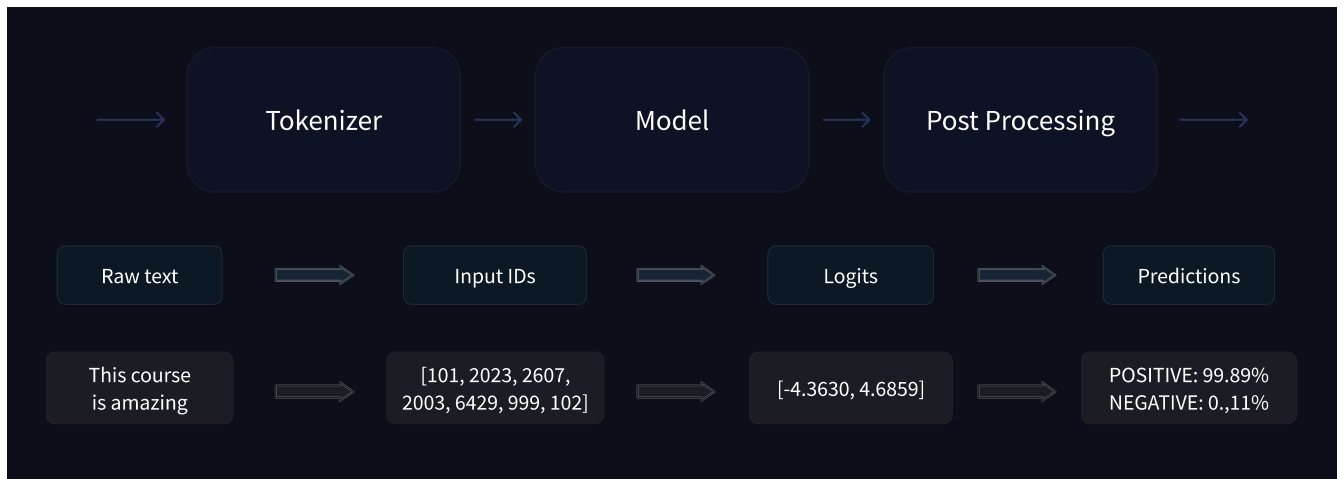


Figure 8. Hugging Face Pipeline [37]

3.2.3 Ανάλυσης Συναισθήματος στο chatbot

Στον κώδικά η συγκεκριμένη βιβλιοθήκη ξεκινάει να χρησιμοποιείτε με την την συνάρτηση `pipeline` της Hugging Face για να δημιουργήσει ένα πίνακα ανάλυσης συναισθήματος χρησιμοποιώντας ένα προεκπαιδευμένο μοντέλο μετασχηματιστή:

```
# Initialize a sentiment analysis pipeline using Hugging Face's Transformers library
sentiment_pipeline = pipeline("sentiment-analysis")
```

`pipeline("sentiment-analysis")`: Αυτή η κλήση της συνάρτησης αρχικοποιεί έναν πίνακα ανάλυσης συναισθήματος χρησιμοποιώντας ένα προεκπαιδευμένο μοντέλο μετασχηματιστή. Το μοντέλο που χρησιμοποιείτε εδώ σχεδιάστηκε για να αναλύει το συναίσθημα (θετικό, αρνητικό, ουδέτερο).

Το αποτέλεσμα που πέρνουμε πρέπει να είναι σε μια συγκεκριμένη κλίμακα και το μοντέλο να ήταν σίγουρο κατά 75% αλλιώς του βάζουμε πως είναι ουδέτερο και έτσι δεν επιρεάζεται το recommendation του χρήστη.

Συνάρτηση `get_emotional_mood(user_input)`:

```
#Analyzes the user's input to determine the emotional mood (uplifting, calm, or neutral) based on the sentiment score.
def get_emotional_mood(user_input):
    result = sentiment_pipeline(user_input)[0]
    if result['label'] == 'POSITIVE' and result['score'] > 0.75:
        return 'uplifting'
    elif result['label'] == 'NEGATIVE' and result['score'] > 0.75:
        return 'calm'
    else:
        return 'neutral'
```

`sentiment_pipeline(user_input)`: Αυτή η γραμμή εκτελεί το pipeline ανάλυσης συναισθήματος που δημιουργήσατε προηγουμένως. Αναλύει το `user_input` (το οποίο είναι μια κειμενική είσοδος που παρέχεται από τον χρήστη) και επιστρέφει ένα αποτέλεσμα.

`result['label']` και `result['score']`: Αυτά τα στοιχεία παρέχουν αντίστοιχα την ετικέτα συναισθήματος που προβλέπεται από το μοντέλο (POSITIVE, NEGATIVE, ή NEUTRAL) και το σκορ που σχετίζεται με αυτήν την πρόβλεψη.

Προσδιορισμός Συναισθηματικής Διάθεσης:

- Αν η ανάλυση συναισθήματος προβλέψει θετικό συναίσθημα με σκορ άνω του 0.75, η συνάρτηση προσδιορίζει τη διάθεση ως 'uplifting'.
- Αν η ανάλυση συναισθήματος προβλέψει αρνητικό συναίσθημα με σκορ άνω του 0.

3.3 Spotify API Integration

Περίληψη API

Το Spotify API επιτρέπει στους προγραμματιστές να αλληλεπιδρούν με τα δεδομένα

και τις λειτουργίες του Spotify μέσω προγραμματιστικών διεπαφών. Το API παρέχει πρόσβαση σε πληροφορίες για τραγούδια, άλμπουμ, καλλιτέχνες, playlists, καθώς και δυνατότητες αναζήτησης και streaming μουσικής. Οι προγραμματιστές μπορούν να χρησιμοποιήσουν το API για να δημιουργήσουν εφαρμογές που ενσωματώνουν μουσικές λειτουργίες του Spotify.

Η πρόσβαση στο Spotify API απαιτεί τη δημιουργία ενός λογαριασμού προγραμματιστή στο Spotify και τη λήψη διαπιστευτηρίων (Client ID και Client Secret) που χρησιμοποιούνται για να αυθεντικοποιήσουν τις αιτήσεις προς το API.

Έτσι στον κώδικα το API του spotify χρησιμοποιείτε ως ο βασικός τρόπος για να γίνεται το recommendation στον χρήστη. Η βασική ιδέα είναι πως ο χρήστης θα μπορεί να ρωτάει κάτι όπως 'Μου αρέσει το X τραγούδι' και το chatbot θα μπορεί να το αναγνωρίζει και να ψάχνει την βάση δεδομένων για κατάλληλα αποτελέσματα.

Κώδικας Για το Spotify API

Ρύθμιση Χρήστη

Αρχικά πρέπει να γίνει μια ρύθμιση ώστε να υπάρχει πρόσβαση στην βάση. Αυτό γίνεται με τα εξής διαπιστευτήρια:

```
from spotipy.oauth2 import SpotifyClientCredentials
import spotipy
```

```
# Sets up the Spotify API client using provided credentials
def setup_spotify_client(client_id, client_secret):
    manager = SpotifyClientCredentials(client_id=client_id, client_secret=client_secret)
    return spotipy.Spotify(client_credentials_manager=manager)
```

Αυτός ο κώδικας δημιουργεί ένα αντικείμενο SpotifyClientCredentials με το client_id και το client_secret και στη συνέχεια επιστρέφει ένα αντικείμενο Spotify που

χρησιμοποιείται για να εκτελέσει αιτήσεις στο API.

Το `client_id` και `client_secret` είναι τα βασικά στοιχεία που χρειάζεται το `spotify` για να μας δώσει πρόσβαση στο `api`. Αυτό θα γινόταν αυτόματα αν μπορούσε να λειτουργήσει το `log in` στο `colab` αλλά λόγω προβλημάτων είναι απαραίτητο να δώσουμε ένα προκαθορισμένο `id`.

Λήψη των Διαθέσιμων Ειδών Μουσικής

Η συνάρτηση `get_spotify_genres(sp)` ανακτά τα διαθέσιμα `seeds` των ειδών (`music genre seeds`) από το `Spotify`. Αυτά τα `seeds` αντιπροσωπεύουν διαθέσιμα είδη μουσικής που μπορούν να χρησιμοποιηθούν ως είσοδοι για τα ερωτήματα συστάσεων. Αυτή η πληροφορία είναι απαραίτητη για να ορίσουν οι χρήστες το είδος μουσικής που τους ενδιαφέρει.

```
# Gets music recommendations from Spotify based on genre and mood-specific attributes
def get_spotify_genres(sp):
    return sp.recommendation_genre_seeds()['genres']
```

Λήψη Μουσικών Συστάσεων

Η συνάρτηση `get_spotify_recommendations` λαμβάνει συστάσεις μουσικής από το `Spotify` βασισμένη σε ένα συγκεκριμένο είδος και επιπλέον χαρακτηριστικά διάθεσης (`valence` και `energy`). Αυτά τα στοιχεία τα πέρνουμε με την βοήθεια του προηγούμενου `pipeline` που αναφέραμε. Αν το `pipeline` έδωσε στο `input` το `tag uplifting` του θέτει το `valence` ως 0.7 και το `energy` ως 0.7 τα οποία είναι και τα δύο αρκετά υψηλά (πέρνουν τιμές 0-1) και αυτό θα δώσει στον χρήστη ένα πιο χαρούμενο τραγούδι. Αν όμως έδωσε το `tag calm` το `valence` και το `energy` θα είναι 0.4 και 0.3 αντιστοίχα. Τέλος αν το μοντέλο δεν ήταν σίγουρο δεν θα δώσει καμιά τιμή και έτσι δεν θα επηρεαστεί το `recommendation` του χρήστη.

```
def get_spotify_recommendations(sp, genre, mood=None):
    attributes = {
        'uplifting': {'valence': 0.7, 'energy': 0.7},
        'calm': {'valence': 0.4, 'energy': 0.3},
        'neutral': {}
    }
    mood_attributes = attributes.get(mood, {})
    results = sp.recommendations(seed_genres=[genre], limit=5, **mood_attributes)

    return format_spotify_results(results)
```

Μορφοποίηση Αποτελεσμάτων από το Spotify

Λόγο προβλημάτων με το google colab δεν ήταν εφικτή η δημιουργία ενός καλύτερου ui για τον χρήστη και έτσι απλώς έγινε μια πιο απλή μορφοποίηση των αποτελεσμάτων.

Η συνάρτηση `format_spotify_results` μορφοποιεί τα αποτελέσματα που λαμβάνονται από το API σε μια πιο αναγνώσιμη μορφή. Όλες οι συναρτήσεις που προτίνουν τραγούδια θα περάσουν από εδώ για να είναι λίγο πιο εύκολο στην κατανόηση το τελικό αποτέλεσμα:

```
Detected genre: happy
Bot: 1. #thatPOWER by will.i.am
Preview not available
Spotify Link: https://open.spotify.com/track/6LuHcIZmMG5LU2hiqxscNb
2. Can't Hold Us (feat. Ray Dalton) by Macklemore & Ryan Lewis
Preview: https://p.scdn.co/mp3-preview/989bcb16f82e4cfcfbefbdf8cf0dd91abe86c84d1?cid=5da9d6e2b4be4c56be0694e7ccb50201
Spotify Link: https://open.spotify.com/track/3bidbhp0Yev4knp8AIu8Xn
3. One Way or Another (Teenage Kicks) by One Direction
Preview: https://p.scdn.co/mp3-preview/53d20635a9c1b890fedf854d6a3efa7030072106?cid=5da9d6e2b4be4c56be0694e7ccb50201
Spotify Link: https://open.spotify.com/track/3jZomv0Ba5qfT05HkqI1p5
4. Paper Planes by M.I.A.
Preview not available
Spotify Link: https://open.spotify.com/track/1kusepF3AacIEtUTYrw4GV
5. Amnesia by 5 Seconds of Summer
Preview not available
Spotify Link: https://open.spotify.com/track/1JCCdiru7fhst0IF4N7WJC
```

Αρχικοποίηση:

```
recommendations = []
```

Αρχικοποίηση μιας αδειας λίστας που θα πάνε τα formatted τραγούδια και θα αποθήκευτον.

Επανάληψη μέσω των Αποτελεσμάτων:

```
for idx, track in enumerate(results['tracks']):
```

Επαναλαμβάνει κάθε κομμάτι στη λίστα results['tracks']. Το idx είναι ο δείκτης του τρέχοντος κομματιού και track αναπαριστά κάθε ξεχωριστό λεξικό κομματιού.

Ανάκτηση Πληροφοριών Καλλιτέχνη:

```
artist = track['artists'][0]['name']
```

Ανακτά το όνομα του καλλιτέχνη για το τρέχον κομμάτι. track['artists'] είναι μια λίστα, και [0] προσπελαύνει τον πρώτο καλλιτέχνη στη λίστα. ['name'] ανακτά το όνομα του καλλιτέχνη.

```
# Formats the api results into more readable outputs
def format_spotify_results(results):
    recommendations = []
    for idx, track in enumerate(results['tracks']):
        artist = track['artists'][0]['name']
        preview_url = track.get('preview_url')
        track_info = f"{idx+1}. {track['name']} by {artist}"
        track_info += f"\nPreview: {preview_url}" if preview_url else "\nPreview not available"
        track_info += f"\nSpotify Link: {track['external_urls']['spotify']}"
        recommendations.append(track_info)
    return "\n".join(recommendations)
```

Χειρισμός URL Προεπισκόπησης/Preview:

```
preview_url = track.get('preview_url')
```

Προσπαθεί να ανακτήσει το URL προεπισκόπησης του κομματιού χρησιμοποιώντας

`track.get('preview_url')`. Αν ένα URL προεπισκόπησης είναι διαθέσιμο, ανατίθεται στη μεταβλητή `preview_url`, διαφορετικά παραμένει `None`.

Δημιουργία Συμβολοσειράς Πληροφοριών Κομματιού:

```
track_info = f"{idx+1}. {track['name']} by {artist}"
```

Κατασκευάζει μια συμβολοσειρά `track_info` που περιλαμβάνει τον αριθμό του κομματιού (`idx+1`), το όνομα του κομματιού (`track['name']`), και το όνομα του καλλιτέχνη (`artist`).

Προσθήκη Πληροφοριών Προεπισκόπησης και συνδέσμου spotify:

Προσθέτει πληροφορίες σχετικά με το URL προεπισκόπησης στην `track_info`. Εάν υπάρχει το `preview_url`, προσθέτει το URL· αλλιώς, υποδεικνύει ότι η προεπισκόπηση δεν είναι διαθέσιμη.

Επίσης προσθέτει τον σύνδεσμο του κομματιού στο Spotify στην `track_info`. `track['external_urls']['spotify']` ανακτά το URL από το λεξικό του κομματιού.

```
track_info += f"\nPreview: {preview_url}" if preview_url else "\nPreview not available"
track_info += f"\nSpotify Link: {track['external_urls']['spotify']}"
```

Προσθήκη στη Λίστα Προτάσεων και Ενοποίηση των Αποτελεσμάτων:

```
recommendations.append(track_info)
return "\n".join(recommendations)
```

Προσθέτει τη μορφοποιημένη συμβολοσειρά `track_info` στη λίστα `recommendations` και Ενώνει όλα τα στοιχεία στη λίστα `recommendations` σε ένα μόνο string χωρισμένο με χαρακτήρες νέας γραμμής (`\n`). Αυτό το string περιέχει τις μορφοποιημένες

πληροφορίες για όλα τα κομμάτια.

Λήψη Παρόμοιων Τραγουδιών

Η συνάρτηση `get_similar_songs` καλύπτει αν ο χρήστης μας δώσει κάποιο πιθανό τραγούδι που του αρέσει:

```
# Provides recommendations of similar songs based on a given track ID
def get_similar_songs(sp, liked_song_id):
    results = sp.recommendations(seed_tracks=[liked_song_id], limit=5)
    tracks = results['tracks']
    recommendations = []
    for idx, track in enumerate(tracks):
        track_name = track['name']
        artist_name = track['artists'][0]['name']
        track_url = track['external_urls']['spotify']
        preview_url = track.get('preview_url')

        recommendation_text = f"{idx+1}. {track_name} by {artist_name}\n"
        if preview_url:
            recommendation_text += f"Preview: {preview_url}\n"
            recommendation_text += f"Spotify Link: {track_url}\n"

        recommendations.append(recommendation_text)

    return "\n".join(recommendations)
```

Η συνάρτηση αυτή πέρνει σαν παραμέτρους το 'sp' που είναι το instance του χρήστη με τους κωδικούς του και ένα id με ένα πιθανό τραγούδι που μπορεί να πρότινε.

```
# Provides recommendations of similar songs based on a given track ID
def get_similar_songs(sp, liked_song_id):
```

Αφού έχουμε ένα id από τραγούδι μπορούμε να χρησιμοποιήσουμε την εντολή `sp.recommendations` για να μας βρεί παρόμοια τραγούδια (μεγιστος αριθμός 5 αν και μπορούμε να το αυξήσουμε). Αυτά τα τραγούδια τα βάζουμε σε μια λίστα με όνομα `recommendations`.

```
results = sp.recommendations(seed_tracks=[liked_song_id], limit=5)
tracks = results['tracks']
recommendations = []
```

Η επόμενη λίγα γραμμές ελέγχουν ένα-ένα τα προτεινόμενα τραγούδια με σκοπό να πάρουν πληροφορίες όπως όνομα καλλιτέχνη, όνομα τραγουδιού και άλλα και να τα βάλουν στην λίστα recommendation.

Τέλος τα βάζει όλα τα strings σε μια λίστα και την επιστρέφει

```
for idx, track in enumerate(tracks):
    track_name = track['name']
    artist_name = track['artists'][0]['name']
    track_url = track['external_urls']['spotify']
    preview_url = track.get('preview_url')

    recommendation_text = f"{idx+1}. {track_name} by {artist_name}\n"
    if preview_url:
        recommendation_text += f"Preview: {preview_url}\n"
    recommendation_text += f"Spotify Link: {track_url}\n"

    recommendations.append(recommendation_text)

return "\n".join(recommendations)
```

Εξαγωγή ονόματος από πρόταση

Αυτή η πρόταση έχει ως σκοπό να πάρουμε/ελεγχουμε το όνομα του τραγουδιού που μπορεί να μας έχει δώσει ο χρήστης. Σαν όρισμα ξεκινάει μόνο με την πρόταση του χρήστη.

```
# Extract the song name from a sentence using regex patterns
def extract_song_name(sentence):
```

Για λίγο πιο πολύ σιγουριά, μιας και ο χρήστης μπορεί να έχει δώσει με πολλούς τρόπους μια πρόταση, υπάρχουν αρκετά regex για να δοκιμασουν την πρόταση. Φυσικά με τον τωρινό τρόπο είναι εύκολο να βρεθεί κάποια περίπτωση που δεν θα μπορέσει να βρεθεί το τραγούδι του χρήστη αλλά είναι ένας αρκετά καλός τρόπος για τώρα. Με περισσότερο χρόνο θα ήταν δυνατή η εύρεση κάποιου τρόπου που να είναι καλύτερος.

```
patterns = [
    re.compile(r"similar to ['\"]?(.*?)['\"]?(?: by |$)", re.IGNORECASE),
    re.compile(r"like ['\"]?(.*?)['\"]?(?: by |$)", re.IGNORECASE),
    re.compile(r"song ['\"]?(.*?)['\"]?(?: by |$)", re.IGNORECASE),
    re.compile(r"https://open.spotify.com/track/([a-zA-Z0-9]+)", re.IGNORECASE)
]
```

Αν βρεθεί κάποιο match στα regex αφερονται οι τρυγύρω λέξεις και το επιστρέφουμε για να σταλθεί στις υπόλοιπες συναρτήσεις. Αν δεν βρεθεί τίποτα στέλνεται πίσω None.

```
for pattern in patterns:
    match = pattern.search(sentence)
    if match:
        return match.group(1).strip()
return None
```

Εξαγωγή Spotify ID από URL

Συμπληρωματική συνάρτηση τις προηγούμενης που είναι για περιπτώσεις που είναι με διαφορετικό τρόπο το url. Αν και θα μπορούσε να είναι στο ίδιο, για κάποιο λόγο δεν λειτουργεί και παραμένει ως bug. Με αυτό τον τρόπο όμως λειτουργεί. Απλώς ελέγχει με διαφορετικό τρόπο αν είναι link αυτό που έδωσε ο χρήστης.

```
# Extracts the Spotify track ID from a URL :D imagine
def extract_spotify_id_from_url(url):
    match = re.search(r"track/([a-zA-Z0-9]+)", url)
    if match:
        return match.group(1)
    return None
```

Finding Song ID by Name

Αυτή η συνάρτηση καλείτε αν βρει το chatbot πως ο χρήστης έδωσε όνομα τραγουδιού μέσω της ανάλυσης που κάνουμε. Έτσι αυτή η συνάρτηση έχει ως ορίσματα το 'sp' και το όνομα του τραγουδιού.

```
# Find a song ID based on its name using the Spotify API
def find_song_id_by_name(sp, song_name):
```

Αρχίζει και ψάχνει να βρει τραγούδια από το spotify που να ταιριάζουν το όνομα με αυτό που έδωσε ο χρήστης . Αφού βρει το τραγούδι που ταιριάζει παίρνει την λίστα με το τραγούδι και το επιστρέφει. Σε περίπτωση που δει πως δεν υπάρχει κάποιο τραγούδι που να ταιριάζει επιστρέφει None.

```
    results = sp.search(q=song_name, type='track', limit=1)
    tracks = results['tracks']['items']
    if tracks:
        return tracks[0]['id'] # Return the Spotify ID of the first match
    else:
        return None
```

3.4 Response

Τώρα που έχουν αναφερθεί οι περισσότερες συναρτήσεις μπορούμε να μιλήσουμε και για την response. Η πιο σημαντική συνάρτηση του chatbot, είναι ένα κεντρικό hub από το οποίο περνάνε όλες οι πληροφορίες και πηγαίνουν στις διαφορετικές συναρτήσεις. Φυσικά ξεκινάει με την δήλωσή της που χρειάζεται την πρόταση του χρήστη, τα δεδομένα του χρήστη για το spotify api, ένα id για το session και ένα object για την διαχείριση του session.

```
def response(sentence, sp, session_id, chat_session):
```

Ανάκτηση Συνεδρίας: Ανακτά την τρέχουσα συνεδρία χρησιμοποιώντας το 'session_id' από το chat_session. Αυτό το αντικείμενο περιέχει κάποιες πληροφορίες της κατάστασης που είναι συγκεκριμένες για την τρέχουσα αλληλεπίδραση του συγκεκριμένου χρήστη. Μπορεί να χρησιμοποιηθεί και στο μέλλον για βελτίωση των δυνατοτήτων του κώδικα.

```
session = chat_session.get_session(session_id)
```

Όπως έχει αναφερθεί χρησιμοποιεί τη συνάρτηση get_emotional_mood για να καθορίσει την συναισθηματική κατάσταση του χρήστη από την είσοδο sentence. Αυτή η διάθεση αποθηκεύεται στη συνεδρία και εκτυπώνεται για σκοπούς εντοπισμού σφαλμάτων. Χρησιμοποιεί την συνάρτηση get_emotional_mood που αναλήφθηκε πιο πριν.

```
emotional_mood = get_emotional_mood(sentence)
session['emotional_mood'] = emotional_mood
print(f"Detected mood: {emotional_mood.capitalize()}")
```

Εξαγωγή Είδους Μουσικής και Ονόματος Τραγουδιού:

- genre_match: Αναζητά ένα είδος μουσικής που αναφέρεται στην πρόταση συγκρίνοντας με μια λίστα από είδη μουσικής του Spotify που ανακτώνται με

τη χρήση του `get_spotify_genres`.

- `song_name`: Εξάγει το όνομα του τραγουδιού από την πρόταση του χρήστη χρησιμοποιώντας τη συνάρτηση `extract_song_name`.

```
genre_match = next((genre for genre in get_spotify_genres(sp) if genre in sentence.lower()), None)
song_name = extract_song_name(sentence)
```

Διαχείριση Αιτήσεων για Παρόμοια Τραγούδια:

- Ελέγχει αν ο χρήστης ζητά παρόμοια τραγούδια ("something similar").
- Αν παρέχεται URL του Spotify, εξάγει το ID του τραγουδιού και ανακτά παρόμοια τραγούδια χρησιμοποιώντας το `get_similar_songs`.
- Αν δεν υπάρχει URL, προσπαθεί να εξάγει το όνομα του τραγουδιού και να βρει το ID του χρησιμοποιώντας το `find_song_id_by_name`.
- Επιστρέφει κατάλληλες απαντήσεις με βάση το αν το ID του τραγουδιού βρέθηκε επιτυχώς.

```
if 'something similar' in sentence:
    if 'spotify.com' in sentence:
        song_id = extract_spotify_id_from_url(sentence)
        if song_id:
            similar_songs = get_similar_songs(sp, song_id)
            return similar_songs
        else:
            return "I couldn't find that song in the link provided. Could you check the link?"
    else:
        song_name = extract_song_name(sentence)
        if song_name:
            if song_name.startswith('https://'):
                song_id = song_name.split('/')[-1].split('?')[0]
            else:
                song_id = find_song_id_by_name(sp, song_name)

            if song_id:
                similar_songs = get_similar_songs(sp, song_id)
                return similar_songs
            else:
                return "I couldn't find that song. Could you try another song name or check the Spotify link?"
        else:
            return "Could you specify which song you'd like something similar to?"
```

Ταξινόμηση Προθέσεων:

- Ταξινομεί την πρόταση του χρήστη για να καθορίσει την πρόθεσή του χρησιμοποιώντας τη συνάρτηση classify.
- Αποθηκεύει την ανιχνευθείσα πρόθεση στη συνεδρία για μελλοντική αναφορά.

```
results = classify(sentence)
intent = None
if results:
    intent = results[0][0]
    session['last_intent'] = intent
```

Διαχείριση Πρόθεσης - Παρόμοια Τραγούδια:

- Διαχειρίζεται αιτήσεις με πρόθεση εύρεσης παρόμοιων τραγουδιών, παρόμοια με το προηγούμενο μπλοκ, αλλά με βάση την ταξινομημένη πρόθεση αντί για άμεσο έλεγχο λέξεων-κλειδιών.

```
if 'similar_songs' in intent:
    song_name = extract_song_name(sentence)
    if song_name:
        song_id = find_song_id_by_name(sp, song_name)
        if song_id:
            return get_similar_songs(sp, song_id)
        else:
            return "I couldn't find that song. Could you try another song name or check the Spotify link?"
    else:
        return "Could you specify which song you'd like something similar to?"
```

Διαχείριση Πρόθεσης - Περισσότερες Προτάσεις:

- Ελέγχει αν ο χρήστης ζητά περισσότερες προτάσεις ή την τελευταία και ανακτά την τελευταία πρόταση χρησιμοποιώντας το `get_last_song_recommendation`.

```
if 'more' in sentence.lower() or 'last one' in sentence.lower():
    return get_last_song_recommendation(session, sp)
```

Διαχείριση Πρόθεσης - Μουσική Πρόταση:

- Διαχειρίζεται αιτήσεις για μουσικές προτάσεις με βάση την ανιχνευθείσα πρόθεση ή τα αναφερόμενα είδη μουσικής.
- Ανακτά προτάσεις χρησιμοποιώντας το `get_spotify_recommendations` βάσει του είδους μουσικής ή της συναισθηματικής διάθεσης.
- Αποθηκεύει και επιστρέφει τις προτάσεις.

```

if 'music_recommendation' in intent or any(genre in sentence.lower() for genre in get_spotify_genres(sp)):
    matched_genre = next((genre for genre in get_spotify_genres(sp) if genre.lower() in sentence.lower()), None)
    if matched_genre:
        session['last_genre'] = matched_genre
        print(f"Detected genre: {matched_genre}")
        recommendations = get_spotify_recommendations(sp, matched_genre, mood=session['emotional_mood'])
        session['last_recommendation'] = recommendations
        return recommendations
    else:
        if emotional_mood:
            recommendations = get_spotify_recommendations(sp, emotional_mood)
            return recommendations
        return "Can you specify a genre? I'm not sure which type of music to suggest based on your mood."

```

Απαντήσεις Προεπιλεγμένων Προθέσεων:

- Διατρέχει προκαθορισμένες προθέσεις και επιστρέφει μια τυχαία απάντηση αν η πρόθεση ταιριάζει.

```

for i in intents['intents']:
    if i['tag'] == intent:
        return random.choice(i['responses'])

```

Εφεδρικές Απαντήσεις:

- Παρέχει εφεδρικά μηνύματα αν δεν ανιχνευθεί συγκεκριμένη πρόθεση ή αν το αίτημα είναι ασαφές.

```

if intent is None:
    return "Can you please specify a bit more? What type of music or which genre are you interested in?"

return "Sorry, I didn't understand that. Can you rephrase or ask something else about music?"

```

3.5 ChatSession Class

Χρησιμοποιήστε για να κρατάει πληροφορίες απο προηγούμενες εντολές του χρήστη,

από εδώ η response και οι άλλες συναρτήσεις πέρνουν το sessions_id. Πιο συγκεκριμένα περιέχει τα : 'history' που είναι πιο παλιές εντολές, 'last_intent' που είναι σε περίπτωση που ο χρήστης ζητήσει κάτι σαν τα προηγούμενα να έχουμε προηγούμενες πληροφορίες και τέλος το 'preference' που δύστηχως δεν είναι τελείως ολοκληρωμένο και θα είναι σε μελλοντική version του κώδικα.

```
# Session Manager for Contextual Awareness
# Manages individual user sessions, storing their interaction history, last detected intent, and preferences
class ChatSession:
    def __init__(self):
        self.sessions = {}

    def get_session(self, session_id):
        if session_id not in self.sessions:
            self.sessions[session_id] = {'history': [], 'last_intent': None, 'preferences': {}}
        return self.sessions[session_id]

    def update_session(self, session_id, key, value):
        session = self.get_session(session_id)
        session[key] = value
```

clean_up_sentence

Βασική συνάρτηση για το nlp κομματι του κώδικα. Έχει κάποια βασικά βήματα που ακολουθεί :

word tokenization : “κόβει” την πρόταση σε λέξεις ώστε να μπορούμε να το χειριστούμε καλύτερα.

Word lemmatization : Βρίσκει τις ρύζες των λέξεων (λήμματα) και τα κάνει όλα πεζά

```
# Tokenizes and lemmatizes a sentence, and generates bigrams
def clean_up_sentence(sentence):
    sentence_words = word_tokenize(sentence) # Tokenize the sentence
    sentence_words = [lemmatizer.lemmatize(word.lower()) for word in sentence_words] # Lemmatize each word
    # Generate bigrams from the list of words
    sentence_bigrams = [' '.join(bigram) for bigram in nltk_bigrams(sentence_words)]
    return sentence_words + sentence_bigrams # Return a list of words and bigrams
```

Μελλοντικές Βελτιώσεις για το Chatbot και Προβλήματα

Δημιουργία Διεπαφής Χρήστη (UI)

Η προσθήκη μιας πιο φιλικής και διαισθητικής διεπαφής χρήστη θα μπορούσε να βελτιώσει σημαντικά την αλληλεπίδραση με το chatbot. Μερικές ιδέες για μελλοντικές βελτιώσεις περιλαμβάνουν:

- **Σχεδιασμός Web UI:** Χρησιμοποιώντας πλαίσια όπως το React ή το Vue.js η κάποιον άλλων βιβλιοθηκών όπως flask και anvil θα μπορούσε να δημιουργηθεί μια διεπαφή που επιτρέπει στους χρήστες να αλληλεπιδρούν με το chatbot μέσω μιας ιστοσελίδας.
- **Mobile App:** Δημιουργία μιας εφαρμογής για κινητά τηλέφωνα, διαθέσιμη σε iOS και Android, χρησιμοποιώντας πλαίσια όπως το React Native ή το Flutter.
- **Προσαρμοσμένο Interface:** Αν όντως πραγματοποιούνταν κάτι τέτοιο φυσικά και θα μπορούσε να ειπάρχει μια παροχή δυνατοτήτων προσαρμογής, όπως αλλαγή θεμάτων, ρυθμίσεις ειδοποιήσεων και προσωπικοποίηση της εμφάνισης του chatbot.

Αν και έγινε προσπάθεια για δημιουργία ενός Web UI με την χρήση flask δυστηχως δεν ήταν δυνατή η δημιουργία του λόγω προβλημάτων με το google colab. Στο μέλλον μπορεί να γίνει μια μετάφορα σε local hosting και να είναι πιο εύκολη η δημιουργία κάτι τέτοιου.

Ενσωμάτωση με το Spotify

Η δυνατότητα προσθήκης τραγουδιών στον λογαριασμό Spotify του χρήστη απευθείας από το chatbot θα μπορούσε να βελτιώσει σημαντικά την εμπειρία χρήστη.

- **Spotify API Integration:** Εκτός απο το recommendation που ήδη γίνεται θα μπορούσε το api του Spotify να χρησιμοποιηθεί και το Spotify Web API για να επιτρέψετε στους χρήστες να συνδέσουν τους λογαριασμούς τους και να προσθέτουν τραγούδια, playlists ή να δημιουργούν νέες λίστες αναπαραγωγής.
- **Αναγνώριση Προθέσεων:** Επίσης θα μπορούσε να βελτιωθεί η

κατηγοριοποίηση προθέσεων για να γίνει εφυκτή η κατανόηση εντολών όπως "Προσθήκη αυτού του τραγουδιού στη λίστα μου στο Spotify".

Βελτιωμένες Απαντήσεις με Χρήση Προηγμένων Μεθόδων

Θα μπορούσε να γίνει χρήση πιο προηγμένων μεθόδων για να παρέχετε μεγαλύτερη ποικιλία και ακρίβεια στις απαντήσεις του chatbot.

- **Ενσωμάτωση GPT-3 ή GPT-4:** Θα μπορούσε να γίνει χρήση πιο εξελιγμένων μοντέλων NLP όπως το GPT-3 ή GPT-4 για να βελτιωθεί η κατανόηση και την απόκριση του chatbot σε πιο περίπλοκες ερωτήσεις και διαλόγους.
- **Εκπαίδευση με Μεγαλύτερο Σύνολο Δεδομένων:** Αν και για τον λόγο που δημιουργήθηκε το chatbot δεν κρίθηκε απαραίτητη η εκπαίδευση του σε μεγαλύτερο σύνολο δεδομένων, μια μελλοντική εκθοση θα μπορούσε να το εκμεταλευτει για την βελτίωση του chatbot.

Υποστήριξη περισσότερων γλωσσών

Τώρα το chatbot λειτουργεί μόνο στα αγγλικά. Στο μέλλον θα μπορούσε να προσθέθει και υποστήριξη για πολλές γλώσσες για να επεκτειθεί η προσβασιμότητα του chatbot σε διεθνείς χρήστες.

- **Γλωσσικά Μοντέλα:** Μοντέλα NLP που υποστηρίζουν πολλές γλώσσες, όπως τα πολυγλωσσικά μοντέλα BERT ή mT5.
- **Αυτόματη Μετάφραση:** Υπηρεσίες αυτόματης μετάφρασης που παρέχουν απαντήσεις στη γλώσσα του χρήστη.

Ενσωμάτωση με άλλες Μουσικές Υπηρεσίες

Εκτός απο το spotify θα μπορούσε να γίνει ενσωμάτωση του chatbot με άλλες μουσικές υπηρεσίες όπως το Apple Music, το Amazon Music και το YouTube Music.

Με αυτό τον τρόπο θα μπορούσε να λυθεί κι'όλλας το πρόβλημα που σε κάποια τραγούδια δεν εμφανίζεται preview link.

4. ΣΥΜΠΕΡΑΣΜΑΤΑ ΚΑΙ ΕΠΙΛΟΓΟΣ

4.1.1 Λειτουργικότητα και Χρησιμότητα:

Το αναπτυγμένο chatbot επιδεικνύει αξιόπιστη λειτουργικότητα στην προτεινόμενη μουσική με βάση τις εισόδους των χρηστών. Χρησιμοποιώντας ανάλυση συναισθημάτων με την τεχνολογία της Hugging Face's Transformers, το chatbot προσδιόρισε ακριβώς τη συναισθηματική διάθεση των εισόδων των χρηστών, ταξινομώντας τις ως αναζωογονητικές, ήρεμες ή ουδέτερες συναισθηματικά.

```
Hi! I'm your music chatbot. Type 'quit' to exit.  
You: can you recommend me something similar to this : https://open.spotify.com/track/1UghkbzB1kuFwt2iy4h29Q?si=8951b216afac4686  
Detected mood: Uplifting
```

4.1.2 Αλληλεπίδραση Χρήστη:

Το chatbot προσφέρει μια φιλική διεπαφή με εύκολα κατανοητά μοτίβα αλληλεπίδρασης. Οι χρήστες μπορούν να ζητήσουν συστάσεις μουσικής εύκολα, καθορίζοντας είτε είδη μουσικής είτε προτιμήσεις για παρόμοια τραγούδια. Η ολοκλήρωση με το API του Spotify εμπλουτίζει την εμπειρία του χρήστη παρέχοντας άμεση πρόσβαση σε προεπισκοπήσεις τραγουδιών και εξωτερικούς συνδέσμους, βελτιώνοντας τη διαφάνεια και τη χρηστικότητα.

```
Hi! I'm your music chatbot. Type 'quit' to exit.  
You: hello, whats your name  
Detected mood: Uplifting  
1/1 [=====] - 0s 24ms/step  
Bot: Hey! Melody here. Looking for some music recommendations?
```

4.1.3 Ενσωμάτωση με το API του Spotify:

Η ενσωμάτωση με το API του Spotify είναι καθοριστική για την λειτουργικότητα του chatbot, επιτρέποντας την πραγματοποίηση σε πραγματικό χρόνο προτάσεων μουσικής και λεπτομερών πληροφοριών τραγουδιών. Παρόλο που είναι αξιόπιστη στην εκτέλεση, παρατηρήθηκαν περιστασιακά περιορισμένα API, επηρεάζοντας τους χρόνους απόκρισης και περιορίζοντας την πρόσβαση σε κάποιες λειτουργίες, όπως οι προεπισκοπήσεις κομματιών υπό συγκεκριμένες συνθήκες.

```
Bot: Hey! Melody here. Looking for some music recommendations?  
You: yea, can you give me something close to rock  
Detected mood: Uplifting  
1/1 [=====] - 0s 22ms/step  
Detected genre: rock  
Bot: 1. Crosstown Traffic by Jimi Hendrix  
Preview: https://p.scdn.co/mp3-preview/a0edbf35d67e50f4c818cd4b8f09333e29db1bfe?cid=5da9d6e2b4be4c56be0694e7ccb50201  
Spotify Link: https://open.spotify.com/track/1ntxpzIUbsSizvuAy61TYy
```

4.1.4 NLP και Μοντέλο Μηχανικής Μάθησης:

Η Φυσική Επεξεργασία Γλώσσας (NLP), με τη χρήση NLTK για τον διαχωρισμό και την λημματοποίηση, αποτελεί αποτελεσματική προεπεξεργασία των εισόδων των χρηστών για την εξαγωγή πραγματικών προθέσεων και πλαισίων. Το μοντέλο μηχανικής μάθησης, εκπαιδευμένο με μια αρχιτεκτονική νευρωνικών δικτύων, επιδεικνύει ικανοποιητική απόδοση στην κατηγοριοποίηση των προθέσεων των χρηστών για προτάσεις μουσικής, αν και περαιτέρω βελτιστοποίηση θα μπορούσε να βελτιώσει την ακρίβεια και την ειδικότητα των απαντήσεων.

4.1.5 Κλιμακωσιμότητα και Ανθεκτικότητα:

Η αρχιτεκτονική του chatbot επιδεικνύει δυνατότητες κλιμακωσιμότητας, προσαρμόζοντας διαφορετικές εισόδους χρηστών και προσαρμόζοντας τις συστάσεις βασιζόμενες σε εξελισσόμενες προτιμήσεις και συναφείς σημάνσεις. Ωστόσο, η διατήρηση του πλαισίου συνεδρίας σε αλληλεπιδράσεις παρατεταμένης διάρκειας παραμένει πρόκληση, επηρεάζοντας τη συνέχεια της εμπειρίας χρήστη σε παρατεταμένες συνεδρίες ή σε διαφορετικές πλατφόρμες.

```
You: that was really nice
Detected mood: Uplifting
1/1 [=====] - 0s 22ms/step
Bot: No problem at all! Melody's got your back.
You: can you recommend something similar to the last one?
Detected mood: Uplifting
Bot: 1. The Last Memoir by Subhavani
Preview: https://p.scdn.co/mp3-preview/5cab78f52f78983f6eb693ac8410453e6bef9965?cid=5da9d6e2b4be4c56be0694e7ccb50201
Spotify Link: https://open.spotify.com/track/5dc7NP9ebhmzszEK6nzING

2. Thoughts by SomeMan
Preview: https://p.scdn.co/mp3-preview/931a73f99e87a5ac81ca74eba91b3c89500fd97e?cid=5da9d6e2b4be4c56be0694e7ccb50201
Spotify Link: https://open.spotify.com/track/74y6IZku49W0sYlMcnF7oE

3. Flow by Portal Plan
Preview: https://p.scdn.co/mp3-preview/ac6bea2effa5acec46cd716bdb435ac5911fc063?cid=5da9d6e2b4be4c56be0694e7ccb50201
Spotify Link: https://open.spotify.com/track/1EoPRePRDBhnZz0uMRjPVY
```

4.1.6 Περιορισμοί και Προκλήσεις:

Τεχνικές προκλήσεις περιλαμβάνουν τη βελτιστοποίηση της αποδοτικότητας κλήσης του API για ομαλότερες χρήσεις, την εκλεπτυσμένη ακρίβεια εντοπισμού συναισθηματικής διάθεσης για μείωση των ψευδών θετικών ή αρνητικών, και τη διευθέτηση περιορισμένων ανακοινώσεων δεδομένων από εξωτερικές πηγές που επηρεάζουν τη συνέπεια της πρότασης.

Συμπέρασμα

Συνολικά, το αναπτυγμένο μουσικό chatbot αποτελεί ένα καλό παράδειγμα για τα εξατομικευμένα συστήματα προτάσεων μουσικής, εκμεταλλευόμενο την ανάλυση συναισθημάτων με τη βοήθεια της τεχνολογίας Hugging Face's Transformers και την

ενσωμάτωση API για την παροχή εξατομικευμένων προτάσεων μουσικής σύμφωνα με τις συναισθηματικές καταστάσεις και τις προτιμήσεις των χρηστών. Η πτυχιακή με τον συνδυασμό της AI, της NLP και των τεχνολογιών ροής μουσικής, προσφέρει πολύτιμες ιδέες και συνεισφορές για την ενίσχυση της χρηστικότητας μέσω εξυπνών διασύνδεσης chatbot.

Μελλοντικές κατευθύνσεις έρευνας θα μπορούσαν να επικεντρωθούν στη βελτίωση της ακρίβειας ανάλυσης συναισθημάτων, την ενσωμάτωση επιπλέον υπηρεσιών ροής μουσικής για ευρύτερη πρόσβαση σε περιεχόμενο και την εφαρμογή πιο εξελιγμένων μοντέλων NLP για πιο διακριτικές αλληλεπιδράσεις. Αυτές οι προκρίσεις θα ενισχύσουν περαιτέρω τη χρησιμότητα και την κλιμακωσιμότητα του chatbot σε διάφορα σενάρια χρήστη, ενισχύοντας τον ρόλο του ως καίριου εργαλείου στη σύγχρονη ψηφιακή κατανάλωση μουσικής.

A. ΠΑΡΑΡΤΗΜΑ

A.1 Βιβλιογραφία και ηλεκτρονικές πηγές

- [1] <https://www.coursera.org/articles/what-is-artificial-intelligence>
- [2] <https://www.bosch.com/stories/history-of-artificial-intelligence/>
- [3] <https://www.britannica.com/technology/artificial-intelligence/Alan-Turing-and-the-beginning-of-AI>
- [4] https://en.wikipedia.org/wiki/Turing_test
- [5] <https://www.theguardian.com/technology/2023/jul/25/joseph-weizenbaum-inventor-eliza-chatbot-turned-against-artificial-intelligence-ai>
- [6] <https://www.sciencedirect.com/science/article/abs/pii/S0020737378800492#:~:text=Mycin%20is%20a%20computer-based,for%20patients%20with%20bacterial%20infections.>
- [7] [https://en.wikipedia.org/wiki/NETtalk_\(artificial_neural_network\)](https://en.wikipedia.org/wiki/NETtalk_(artificial_neural_network))
- [8] https://www.researchgate.net/figure/Example-of-ELIZA-ELIZA-a-chatbot-was-designed-by-Joseph-Weizenbaum-to-imitate-a_fig1_348306833
- [9] <https://www.scientificamerican.com/article/20-years-after-deep-blue-how-ai-has-advanced-since-conquering-chess/>

- [10] <https://www.ibm.com/history/watson-jeopardy>
- [11] <https://www.who.int/teams/digital-health-and-innovation/harnessing-artificial-intelligence-for-health>
- [12] <https://www.unesco.org/en/digital-education/artificial-intelligence>
- [13] <https://www.unep.org/news-and-stories/story/how-artificial-intelligence-helping-tackle-environmental-challenges>
- [14] <https://www.oracle.com/chatbots/what-is-a-chatbot/>
- [15] <https://www.apa.org/monitor/2023/07/psychology-embracing-ai#:~:text=URL%3A%20https%3A%2F%2Fwww.apa.org%2Fmonitor%2F2023%2F07%2Fpsychology,100>
- [16] [https://en.wikipedia.org/wiki/ELIZA#/media/File:Video_Game_Museum_in_Berlin_\(44129332940\).jpg](https://en.wikipedia.org/wiki/ELIZA#/media/File:Video_Game_Museum_in_Berlin_(44129332940).jpg)
- [17] <https://www.weforum.org/agenda/2023/05/can-ai-actually-increase-productivity/#:~:text=URL%3A%20https%3A%2F%2Fwww.weforum.org%2Fagenda%2F2023%2F05%2Fcan>
- [18] https://alvinntnu.github.io/NTNU_ENC2045_LECTURES/nlp/nlp-pipeline.html
- [19] https://en.wikipedia.org/wiki/List_of_chatbots
- [20] <https://www.giosg.com/blog/ai-chatbot-vs-rule-based-chatbot>
- [21] https://en.wikipedia.org/wiki/Convolutional_neural_network
- [22] [https://aws.amazon.com/what-is/recurrent-neural-network/#:~:text=A%20recurrent%20neural%20network%20\(RNN,complex%20semantics%20and%20syntax%20rules](https://aws.amazon.com/what-is/recurrent-neural-network/#:~:text=A%20recurrent%20neural%20network%20(RNN,complex%20semantics%20and%20syntax%20rules)
- [23] <https://mindtitan.com/resources/guides/chatbot/how-do-chatbots-work/>
- [24] <https://www.ibm.com/topics/natural-language-processing>
- [25] <https://builtin.com/machine-learning/relu-activation-function#:~:text=What%20does%20ReLU%20stand%20for,as%20the%20rectifier%20activation>

[%20function.](#)

[26] <https://www.simplilearn.com/tutorials/machine-learning-tutorial/what-is-epoch-in-machine-learning#:~:text=Each%20time%20a%20dataset%20passes,training%20data%20through%20the%20algorithm.>

[27] https://en.wikipedia.org/wiki/Bag-of-words_model

[28] <https://arxiv.org/abs/1412.6980>

[29] <https://www.ibm.com/watson>

[30] <https://huggingface.co/docs/hub/index>

[31]

<https://nlp.stanford.edu/~wcmac/papers/20140716-UNLU.pdf><https://medium.com/sciforce/nlp-vs-nlu-from-understanding-a-language-to-its-processing>

[32] <https://rasa.com/>

[33] <https://en.wikipedia.org/wiki/Dialogflow>

[34] <https://deepgram.com/ai-apps/landbot-ai>

[35] <https://www.tensorflow.org/>

[36] <https://developer.spotify.com/documentation/web-api/reference/get-track>

[37] <https://www.nltk.org/howto.html>

[38] <https://huggingface.co/models>

[39] <https://handsonai.medium.com/build-a-chat-bot-from-scratch-using-python-and-tensorflow-fd189bcfae45>

[40] <https://www.projectpro.io/article/python-chatbot-project-learn-to-build-a-chatbot-from-scratch/429>