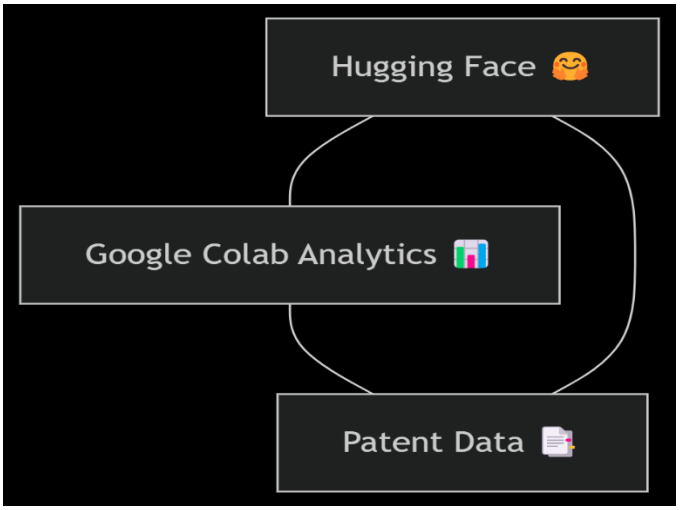
ΔΙΕΘΝΕΣ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΤΗΣ ΕΛΛΑΔΟΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

«Ανάπτυξη Εργαλείων Λογισμικού για Επεξεργασία & Ανάλυση Μεγάλου Όγκου Δεδομένων Πατεντών»



Του φοιτητή
Μυλωνίδη Αναστάσιου
Αρ. Μητρώου: 154510

Επιβλέπων
Σαλαμπάσης Μιχάλης
Βαθμίδα Καθηγητής

Ημερομηνία 28/05/2025

Τίτλος Δ.Ε. Ανάπτυξη Εργαλείων Λογισμικού για Επεξεργασία & Ανάλυση Μεγάλου Όγκου Δεδομένων Πατεντών

Κωδικός Δ.Ε. 25130

Ονοματεπώνυμο φοιτητή Μυλωνίδης Αναστάσιος

Ονοματεπώνυμο εισηγητή Σαλαμπάσης Μιχάλης

Ημερομηνία ανάληψης Δ.Ε. 24-02-2025

Ημερομηνία περάτωσης Δ.Ε. 28-05-2025

Βεβαιώνω ότι είμαι ο συγγραφέας αυτής της εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης, έχω καταγράψει τις όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών, εικόνων και κειμένου, είτε αυτές αναφέρονται ακριβώς είτε παραφρασμένες. Επιπλέον, βεβαιώνω ότι αυτή η εργασία προετοιμάστηκε από εμένα προσωπικά, ειδικά ως διπλωματική εργασία, στο Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του ΔΙ.ΠΑ.Ε.

«Σε αυτούς που είναι δίπλα μου»

Πρόλογος

Η επιλογή του συγκεκριμένου θέματος ως διπλωματική εργασία ήταν επηρεασμένη σε μεγάλο βαθμό από την επαγγελματική ιδιότητα που κατέχω ως DevOps Engineer. Η καθημερινή ενασχόληση με διεργασίες και τεχνολογίες που αποσκοπούν στην ενίσχυση της αποτελεσματικότητας και τη μείωση των σφαλμάτων για την παράδοση λογισμικού μου καλλιέργησαν το ενδιαφέρον για εργαλεία επεξεργασίας και ανάλυσης μεγάλου όγκου δεδομένων. Η σημαντικότητα των Big Data τεχνολογιών είναι πλέον αδιαμφισβήτητη και η ενσωμάτωση τους σε σύγχρονα λειτουργικά pipelines είναι απαραίτητη. Η ανάπτυξη τέτοιων εργαλείων για δεδομένα πατεντών είναι ιδιαίτερα σημαντική καθώς ο συγκεκριμένος τομέας έχει μεγάλη ερευνητική δραστηριότητα αφού οι πατέντες αποτελούν πολύτιμες πηγές τεχνικής και εμπορικής πληροφορίας.

Περίληψη

Στην παρούσα εργασία παρουσιάζεται η ανάπτυξη εργαλείων τα οποία επιτρέπουν την επεξεργασία και ανάλυση μεγάλου όγκου δεδομένων πατεντών. Αρχικά αφού αναλυθούν οι δύο συλλογές πατεντών που χρησιμοποιήθηκαν, η CLEF-IP και η WPI, παρουσιάζεται η μετατροπή του μεγάλου όγκου πρωτογενών δεδομένων των συλλογών σε μικρότερα εξειδικευμένα υποσύνολα. Έπειτα εξετάζεται το πως η πλατφόρμα Hugging Face εξυπηρετεί την οργάνωση, φιλοξενία και διαμοίραση αυτών των υποσυνόλων παρέχοντας παράλληλα και versioning των δεδομένων αλλά και πως αυτά υλοποιήθηκαν για το υποσύνολο της CLEF-IP. Για την φόρτωση των δεδομέων της CLEF-IP από το Hugging Face παρουσιάζεται η ανάπτυξη ενός προσαρμοσμένου πάνω στην δομή του υποσυνόλου loading script σε Python.

Στη συνέχεια παρουσιάζονται μέσα από το Google Colab τα Python scripts που αναπτύχθηκαν για την εξαγωγή αναλυτικών δεδομένων από το υποσύνολο της CLEF-IP και αναλύονται τα αποτελέσματα και οι οπτικοποιήσεις για την εξαγωγή χρήσιμων συμπερασμάτων. Η διαχείριση των δεδομένων στην Python έγινε με την βιβλιοθήκη pandas, ενώ οι οπτικοποιήσεις με τις βιβλιοθήκες matplotlib και plotly.

Τελευταίο και σημαντικό κομμάτι είναι η μεταφορά μιας από τις αναλύσεις αυτούσια σε ένα υποσύνολο της WPI, αφού πρώτα παρουσιαστεί το script για την παραγωγή του από τα πρωτογενή δεδομένα της συλλογής, αποδεικνύοντας την μεταφερσιμότητα και αναπαραγωγιμότητα του κώδικα.

«Development of Software Tools for Processing & Analysis of Patent Big Data»

«Anastasios Mylonidis»

Abstract

In this thesis, we present the development of tools that enable the processing and analysis of patent big data. First, after examining the two patent collections used—CLEF-IP and WPI—we describe how the vast amount of raw collection data was transformed into smaller, specialized subsets. Next, we explore how the Hugging Face platform supports the organization, hosting, and sharing of these subsets, including data versioning, and we show how this was implemented for the CLEF-IP subset.

To load the CLEF-IP data from Hugging Face, we developed a custom Python loading script tailored to the subset's structure. We then demonstrate, via Google Colab, the Python scripts created to extract detailed analytics from the CLEF-IP subset, and we analyze the resulting data and visualizations to draw useful conclusions. Data manipoulation in Python was handled with thepandas library, while visualizations were produced using matplotlib and plotly.

Finally, an important step was transfering one of these analyses intact to a subset of the WPI collection, after presenting the script used to generate it from the raw WPI data,thereby proving the transferability and reproducibility of the code.

Ευχαριστίες

Θα ήθελα πρώτα να ευχαριστήσω την οικογένεια μου για την υποστήριξη καθ' όλη την διάρκεια της φοιτητικής μου πορείας. Ακόμα θα ήθελα να ευχαριστήσω τον επιβλέποντα μου Καθηγητή Κ. Μιχάλη Σαλαμπάση που με την πολύτιμη καθοδήγηση του και τις συμβουλές του οδήγησε στην διεκπεραίωση της εργασίας μου. Τέλος θα ήθελα να ευχαριστήσω την Κ. Ελένη Καματέρη για την άψογη συνεργασία της, το πόσο διαθέσιμη ήταν αλλά και την βοήθεια της σε όλη την διάρκεια της υλοποίησης της εργασίας μου.

Περιεχόμενα

Κατάλογος Εικόνων

Συντομογραφίες

IPC	International Patent Classification
CPC	Cooperative Patent Classification
NLP	Natural Language Processing
CSV	Comma Separated Value
JSON	JavaScript Object Notation
GB	GigaByte
UI	User Interface
XML	Extensible Markup Language
EPO	European Patent Office
USPTO	United States Patent and Trademark Office
WIPO	World Intellectual Property Organization
IPCR	International Patent Classification Reform
ECLA	European Classification System
YAML	YAML Ain't Markup Language
PNG	Portable Network Graphic
JPEG	Joint Photographic Experts Group
WAV	Waveform Audio File Format
TB	TeraByte
RAM	Random Access Memory
GPU	Graphics Processing Unit
HTML	HyperText Markup Language
TPU	Tensor Processing Unit
MB	MegaBytes

Κεφάλαιο 1ο: Εισαγωγή

1.1 Γενικά

Οι πατέντες (ή διπλώματα ευρεσιτεχνίας) αποτελούν έναν από τους σημαντικότερους θεσμούς της πνευματικής ιδιοκτησίας και αποσκοπούν στην προστασία τεχνολογικών εφευρέσεων. Χορηγούνται από κρατικές ή διεθνείς αρχές σε φυσικά ή νομικά πρόσωπα, προσφέροντάς τους αποκλειστικό δικαίωμα εκμετάλλευσης της εφεύρεσης για περιορισμένο χρονικό διάστημα, συνήθως για 20 έτη [1].

Η απονομή ενός τέτοιου δικαιώματος συνιστά μια μορφή ανταμοιβής προς τον εφευρέτη, με αντάλλαγμα τη δημόσια αποκάλυψη της τεχνικής γνώσης που ενσωματώνεται στην εφεύρεση. Αυτή η νομική καινοτομία δημιουργεί έναν μηχανισμό που ενισχύει τη διάχυση της γνώσης και ταυτόχρονα προάγει την καινοτομία, δημιουργώντας οικονομικά κίνητρα για επενδύσεις στην έρευνα και ανάπτυξη (R&D) [2].

Μερικά από τα δομικά στοιχεία μιας τυπικής πατέντας είναι:

- **Αριθμό Δημοσίευσης**
- **Τίτλο** (Title)
- **Περίληψη (Abstract)**
- **Περιγραφή (Description)**
- **Αξιώσεις (Claims)** – το νομικό πλαίσιο της προστασίας
- **Λίστα Αιτούντων**
- **Λίστα Εφευρετών**
- **Ταξινομήσεις (π.χ. IPC, CPC)**

Η ενότητα των claims είναι καθοριστικής σημασίας, καθώς ορίζει με ακρίβεια το εύρος της προστασίας που προσφέρεται από την πατέντα.

Η σημασία των πατεντών ξεπερνά την νομική τους διάσταση. Αποτελούν πολύτιμες πηγές τεχνικής και εμπορικής πληροφορίας, καθώς κάθε πατέντα δημοσιεύει λεπτομερώς μια τεχνολογική λύση. Υπολογίζεται ότι περισσότερο από το 70% της τεχνολογικής πληροφορίας που υπάρχει παγκοσμίως δημοσιεύεται πρώτα σε πατέντες και όχι σε επιστημονικά άρθρα [3].

Η ραγδαία αύξηση του όγκου των αιτήσεων πατεντών τα τελευταία χρόνια έχει μετατρέψει τις βάσεις δεδομένων πατεντών σε πολύτιμες δεξαμενές τεχνολογικής πληροφορίας. Η ανάλυση δεδομένων πατεντών (patent data analysis) προσφέρει δυνατότητες εντοπισμού τεχνολογικών τάσεων, καθώς και στρατηγικών προσεγγίσεων από εταιρείες και ερευνητικούς οργανισμούς [4].

Ωστόσο, η επεξεργασία τέτοιων δεδομένων δεν είναι απλή. Οι πατέντες είναι πολυδιάστατα έγγραφα που περιλαμβάνουν δομημένα πεδία (π.χ., αριθμούς αίτησης, ημερομηνίες), ημιδομημένα πεδία (π.χ., λίστες εφευρετών), αλλά και μη δομημένο κείμενο σε πεδία όπως η περιγραφή (description) και οι αξιώσεις (claims) [5]. Αυτό καθιστά την επεξεργασία τους μια πρόκληση που απαιτεί εξειδικευμένες τεχνικές από το πεδίο της επιστήμης δεδομένων, της φυσικής γλώσσας (NLP), αλλά και της επιστήμης της πληροφορίας.

1.2 Κίνητρο και στόχοι της εργασίας

Στην παρούσα εργασία ο κύριος στόχος είναι να παρουσιαστούν κάποια εργαλεία και μια μεθοδολογία για την επεξεργασία μεγάλου όγκου δεδομένων έτσι ώστε από μια τεράστια πρωτογενή συλλογή δεδομένων πατεντών να παραχθεί ένα εύκολα αναπαραγώγιμο και διαμοιραζόμενο υποσύνολο. Επίσης

ακόμα ένας στόχος είναι τα εργαλεία που θα παρουσιαστούν να παρέχουν την δυνατότητα εξαγωγής αναλυτικών δεδομένων από το εκάστοτε υποσύνολο ώστε να αποδειχθεί η δυνατότητα αξιοποίησης των υποσυνόλων για εξαγωγή χρήσιμων συμπερασμάτων. Τέλος, ένας ακόμα στόχος είναι η μεθοδολογία που θα αναπτυχθεί είναι να μπορεί να μεταφερθεί σε διαφορετικές συλλογές πατεντών και να μην εξυπηρετεί αποκλειστικά της ανάγκες κάποιας συγκεκριμένης συλλογής.

1.3 Δομή της εργασίας

Στο Κεφάλαιο 2 θα γίνει περιγραφή της CLEF-IP και της WPI, δηλαδή των δύο συλλογών των οποίων τα πρωτογενή δεδομένα θα χρησιμοποιηθούν για την παραγωγή υποσυνόλων. Θα παρουσιαστούν οι δομές τους και τα πεδία που περιέχουν και θα γίνει περιγραφή της διαδικασίας δημιουργίας υποσυνόλων αυτών. Στο Κεφάλαιο 3 θα παρουσιαστεί η πλατφόρμα Hugging Face και δυνατότητες που έχει και θα αναλυθεί το πως χρησιμοποιήθηκε ως λύση για την φιλοξενία και διαμοιρασμό των datasets. Στο Κεφάλαιο 4 θα παρουσιαστεί ο κώδικας που αναπτύχθηκε στο Google Colab για την εξαγωγή αναλυτικών δεδομένων στο dataset της CLEF-IP και θα αναλυθούν τα συμπεράσματα που προκύπτουν από τα εξαγόμενα analytics. Στο Κεφάλαιο 5 θα παρουσιαστεί κατά πόσο μεθοδολογία των αναλύσεων του υποσυνόλου της CLEF-IP μπορεί να μεταφερθεί και αν ναι πόσο εύκολα, σε ένα υποσύνολο της WPI. Τέλος, στο Κεφάλαιο 6 θα παρουσιαστούν τα συμπεράσματα που προκύπτουν από την εφαρμογή των εργαλείων και της μεθοδολογίας.

Κεφάλαιο 2ο: Περιγραφή συλλογών

Στο Κεφάλαιο αυτό θα περιγραφούν οι δύο συλλογές, CLEF-IP και WPI, των οποίων τα δεδομένα θα αξιοποιηθούν στην παρούσα εργασία. Για την CLEF-IP θα γίνει πλήρης περιγραφή της και θα εξηγηθεί η ανάγκη για δημιουργία εξειδικευμένου υποσυνόλου καθώς σε αυτό θα γίνει η εξαγωγή των αναλυτικών δεδομένων και η συμπερασματολογία με βάση αυτά. Επίσης θα περιγραφεί η δομή των XML αρχείων που περιέχουν τα πρωτογενή δεδομένα και αλλά και ο τρόπος με τον οποίο έγινε ο καθαρισμός και η εξαγωγή των δεδομένων για το υποσύνολο. Όσον αφορά την WPI, μιας και η αξιοποίηση της στην παρούσα εργασία είναι για την αξιολόγηση της μεταφερσιμότητας των αναλύσεων που θα εξαχθούν στην CLEF-IP, στο Κεφάλαιο αυτό θα γίνει η γενική περιγραφή της συλλογής αλλά και των XML αρχείων με τα πρωτογενή δεδομένα των διπλωμάτων ευρεσιτεχνίας. Ο τρόπος με τον οποίο έγινε η δημιουργία του εξειδικευμένου υποσυνόλου θα αναλυθεί στο Κεφάλαιο 5

2.1 CLEF-IP

2.1.1 Περιγραφή της συλλογής CLEF-IP

Η συλλογή CLEF-IP (Cross-Language Evaluation Forum – Intellectual Property) αποτελεί ένα ολοκληρωμένο και δημόσια διαθέσιμο σύνολο δεδομένων, το οποίο έχει σχεδιαστεί για την αξιολόγηση τεχνικών ανάκτησης πληροφορίας. Παρουσιάστηκε αρχικά στο πλαίσιο των εργαστηρίων αξιολόγησης CLEF-IP, με σκοπό την ενίσχυση της έρευνας σε τομείς όπως η ανάκτηση προγενέστερων σχετικών πατεντών (prior art search), η πολυγλωσσική ανάκτηση πατεντών, η ταξινόμηση πατεντών και άλλες σχετικές εφαρμογές.

Η συλλογή περιλαμβάνει περίπου 3,5 εκατομμύρια έγγραφα διπλωμάτων ευρεσιτεχνίας, που προέρχονται από το Ευρωπαϊκό Γραφείο Διπλωμάτων Ευρεσιτεχνίας (EPO) και καλύπτουν περίοδο από τη δεκαετία του 1970 έως τα τέλη της δεκαετίας του 2000. Κάθε έγγραφο περιλαμβάνει:

- Πλήρες κείμενο του διπλώματος ευρεσιτεχνίας (τίτλος, περίληψη, περιγραφή και αξιώσεις)
- Δομημένα μεταδεδομένα, όπως:
    - Κωδικοί της Διεθνούς Ταξινόμησης Διπλωμάτων Ευρεσιτεχνίας (IPC)
    - Αριθμοί αίτησης και δημοσίευσης
    - Ημερομηνίες κατάθεσης και δημοσίευσης
    - Ονόματα αιτούντων και εφευρετών
    - Αναφορές σε άλλα διπλώματα (citations)
- Πολυγλωσσικό περιεχόμενο, κυρίως σε Αγγλικά, Γαλλικά και Γερμανικά, αντανακλώντας τη τριγλωσσική φύση του EPO

Η συλλογή περιλαμβάνει επίσης πληροφορίες για τις οικογένειες διπλωμάτων ευρεσιτεχνίας, συνδέοντας σχετικά διπλώματα που έχουν κατατεθεί σε διαφορετικές δικαιοδοσίες, καθώς και αναφορές, που είναι ιδιαίτερα χρήσιμα για εργασίες που αφορούν την εκτίμηση νομικής ή τεχνολογικής συνάφειας.

Η συλλογή CLEF-IP έχει χρησιμοποιηθεί εκτενώς στην πειραματική έρευνα στον τομέα της ανάκτησης πληροφορίας και ταξινόμησης πατεντών και προσφέρει ένα απαιτητικό πεδίο δοκιμών, λόγω της τεχνικής πολυπλοκότητας, του μεγάλου όγκου κειμένου και της θεματικής εξειδίκευσης των

εγγράφων. Ο σχεδιασμός της επιτρέπει πολυγλωσσικά πειράματα και είναι ιδιαίτερα πολύτιμη για την αξιολόγηση συστημάτων ανάκτησης και ταξινόμησης σε νομικά-τεχνικά περιβάλλοντα

2.1.2 Ανάγκη δημιουργίας εξειδικευμένου υποσυνόλου για ταξινόμηση

Στη σχετική βιβλιογραφία έχουν προταθεί διάφορες εργασίες ταξινόμησης πατεντών, οι οποίες βασίζονται σε υποσύνολα της συλλογής CLEF-IP. Ωστόσο, κάθε μελέτη φαίνεται να χρησιμοποιεί διαφορετικά αποσπάσματα της συλλογής και με διαφορετική μεθοδολογία επιλογής, χωρίς να υπάρχει μία ενοποιημένη ή τυποποιημένη διαδικασία. Συγκεκριμένα, οι περισσότεροι ερευνητές περιγράφουν με γενικούς όρους τον τρόπο με τον οποίο εξήγαγαν τα δεδομένα για τα πειράματά τους, χωρίς να παρέχουν επαρκείς λεπτομέρειες ή αναπαραγώγιμες οδηγίες.

Για παράδειγμα, αρκετές εργασίες αναφέρουν ότι χρησιμοποιούν «πατέντες που περιέχουν ταξινομητικούς κωδικούς», χωρίς όμως να διευκρινίζεται σε ποιους τύπους κωδικών αναφέρονται. Η συλλογή CLEF-IP περιλαμβάνει διαφορετικά είδη ταξινομήσεων, όπως οι IPC main/further codes, οι IPCR (International Patent Classification Reformatted) καθώς και οι ECLA (European Classification), και η ερμηνεία αυτών μπορεί να επηρεάσει σημαντικά την επιλογή των εγγράφων και των αντίστοιχων ετικετών ταξινόμησης.

Επιπλέον, σπάνια γίνεται σαφές πώς αντιμετωπίζονται περιπτώσεις όπου το ίδιο δίπλωμα ευρεσιτεχνίας εμφανίζεται σε πολλαπλές εκδόσεις (π.χ. διαφορετικά kind codes, όπως A1, A2, B1), οι οποίες ενδέχεται να διαφέρουν ως προς την πληρότητα των πεδίων (π.χ. κάποιο kind code μπορεί να περιλαμβάνει πλήρη κωδικοποίηση IPC, ενώ άλλο όχι). Δεν διευκρινίζεται αν οι συγγραφείς ενοποίησαν τις εκδοχές, επέλεξαν μία συγκεκριμένη (και ποια), ή αφαίρεσαν τέτοιες περιπτώσεις από το σύνολο δεδομένων.

Τέτοιες ασάφειες καθιστούν δύσκολη τη σύγκριση αποτελεσμάτων μεταξύ διαφορετικών μελετών και περιορίζουν τη δυνατότητα αναπαραγωγής των πειραμάτων. Για τον λόγο αυτό, κρίνεται απαραίτητη η καθιέρωση τυποποιημένων πρωτοκόλλων προεπεξεργασίας και τεκμηρίωσης της επιλογής υποσυνόλων από την CLEF-IP, καθώς και η σαφής αποτύπωση των κριτηρίων επιλογής, φιλτραρίσματος και συσχέτισης των δεδομένων.

Μια τέτοια προσπάθεια πραγματοποιήθηκε και στο πλαίσιο της παρούσας διπλωματικής εργασίας. Συγκεκριμένα, δημιουργήθηκε ένα καθαρό και τεκμηριωμένο υποσύνολο της συλλογής CLEF-IP, το οποίο περιλαμβάνει αποκλειστικά πατέντες στην αγγλική γλώσσα που πληρούν συγκεκριμένα κριτήρια πληρότητας και ταξινόμησης. Πιο αναλυτικά, στο υποσύνολο περιλαμβάνονται μόνο εκείνα τα έγγραφα για τα οποία υπάρχουν διαθέσιμα όλα τα κύρια πεδία κειμένου — δηλαδή περίληψη (abstract), περιγραφή (description) και αξιώσεις (claims) — καθώς και τουλάχιστον ένας κύριος κωδικός ταξινόμησης (main IPC code).

Η διαδικασία εξαγωγής και καθαρισμού των δεδομένων περιγράφεται αναλυτικά, με στόχο τη διαφάνεια και την αναπαραγωγιμότητα. Πραγματοποιήθηκε πλήρης ανάλυση της δομής XML των εγγράφων της CLEF-IP, με ειδική μεταχείριση περιπτώσεων πολλαπλών εκδόσεων του ίδιου διπλώματος (διαφορετικά kind codes). Επίσης, εφαρμόστηκαν φίλτρα για την αφαίρεση εγγράφων με ελλιπή μεταδεδομένα.

Το τελικό υποσύνολο, καθώς και ένα αντιπροσωπευτικό δείγμα (sample subset) για άμεσες δοκιμές, έχουν ανέβει στην πλατφόρμα Hugging Face Datasets, καθιστώντας τα άμεσα διαθέσιμα για την ερευνητική κοινότητα. Παράλληλα, έχει δημοσιευθεί ανοιχτά ο πηγαίος κώδικας parsing και

εξαγωγής δεδομένων, ο οποίος περιλαμβάνει όλα τα βήματα επεξεργασίας, σε κατάλληλα σχολιασμένη μορφή. Με τον τρόπο αυτό, επιδιώκεται η διευκόλυνση της χρήσης της συλλογής, η συγκρισιμότητα μεταξύ διαφορετικών προσεγγίσεων και η συμβολή στην αναπαραγώγιμη έρευνα στον τομέα της ανάκτησης και ταξινόμησης πατεντών.

2.1.3 Δομή XML των αρχείων της CLEF-IP

Τα δεδομένα της συλλογής CLEF-IP διατίθενται σε μορφή XML αρχείων, σύμφωνα με το πρότυπο ST.36 του WIPO, το οποίο αποτελεί ένα κοινά αποδεκτό σχήμα για την αναπαράσταση πληροφοριών διπλωμάτων ευρεσιτεχνίας. Κάθε XML αρχείο αντιστοιχεί σε μια συγκεκριμένη εκδοχή (version) ενός διπλώματος ευρεσιτεχνίας και περιλαμβάνει δομημένη πληροφορία που αφορά τόσο τα βιβλιογραφικά δεδομένα, όσο και το τεχνικό περιεχόμενο του εγγράφου.

Η ιεραρχική δομή κάθε XML εγγράφου όπως φαίνεται και στις Εικόνες 2.1 και 2.2 οργανώνεται γύρω από το κύριο στοιχείο <patent-document>, το οποίο συνοδεύεται από χαρακτηριστικά (attributes) όπως:

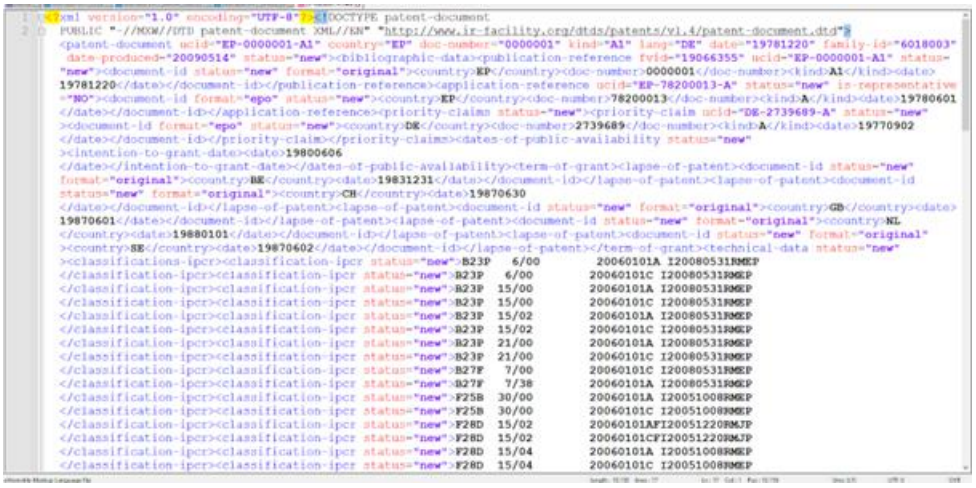
- ucid (unique document identifier)
- doc-number (αριθμός πατέντας)
- kind (είδος εγγράφου π.χ. A1, B2)
- date (ημερομηνία δημοσίευσης)
- country (χώρα προέλευσης)
- lang (γλώσσα εγγράφου)
- date-produced (ημερομηνία παραγωγής της συγκεκριμένης εκδοχής)
- status (κατάσταση του διπλώματος: π.χ. granted, application)

Στο εσωτερικό του εγγράφου, υπάρχουν ξεχωριστά στοιχεία για κάθε ενότητα:

- <bibliographic-data>: περιέχει τα μεταδεδομένα (π.χ. τίτλο, αιτούντες, εφευρέτες, ημερομηνίες κατάθεσης και δημοσίευσης, ταξινομήσεις IPC κ.ά.)
- <invention-title lang="EN">: ο τίτλος της εφεύρεσης στην αγγλική γλώσσα,
- <abstract lang="EN">: η περίληψη του διπλώματος
- <description lang="EN">: το αναλυτικό τεχνικό κείμενο που περιγράφει την εφεύρεση,
- <claims lang="EN">: το σύνολο των αξιώσεων
- <classification-ipcr>: κωδικοί IPC (International Patent Classification)
- <main-classification> και <further-classification>: κύρια και επιπλέον ταξινομήσεις, αντίστοιχα
- <classification-symbol>: χρήση ECLA (European Classification)
- <applicant> και <inventor>: στοιχεία των φυσικών ή νομικών προσώπων που υπέβαλαν ή εφηύραν την πατέντα.

Μια σημαντική ιδιαιτερότητα είναι ότι πολλές πατέντες υπάρχουν σε περισσότερες από μία εκδοχές, με διαφορετικά kind codes, κάτι που απαιτεί προσοχή κατά την ενοποίηση ή την επιλογή της πιο πρόσφατης/πληρέστερης έκδοσης. Επίσης, παρόλο που τα περισσότερα πεδία υπάρχουν σε κάθε XML,

η πληρότητα των στοιχείων διαφέρει, γεγονός που καθιστά απαραίτητο τον έλεγχο ύπαρξης συγκεκριμένων γλωσσικών εκδόσεων (συνήθως EN) και ενοτήτων.



Εικόνα 2.1: Δομή XML αρχείο συλλογής CLEF-IP (μέρος α)

Εικόνα 2.2: Δομή XML αρχείο συλλογής CLEF-IP (μέρος β)

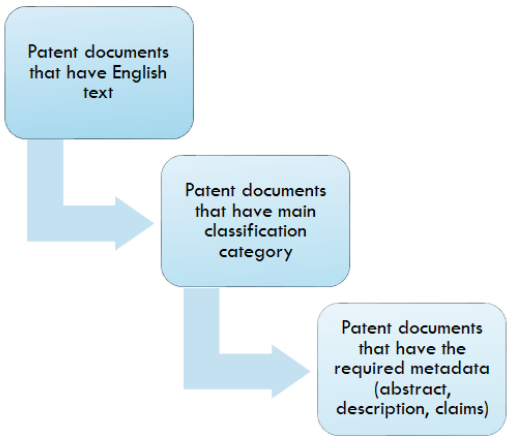
2.1.4 Εξαγωγή και καθαρισμός των δεδομένων

Στο πλαίσιο της παρούσας εργασίας, πραγματοποιήθηκε μια αναλυτική και δομημένη διαδικασία εξαγωγής και καθαρισμού των δεδομένων της συλλογής CLEF-IP, με στόχο τη δημιουργία ενός αξιόπιστου και πλήρως τεκμηριωμένου υποσυνόλου δεδομένων.

Αρχικά, όλα τα πρωτογενή δεδομένα της συλλογής, τα οποία διατίθενται σε μορφή XML αρχείων, εισήχθησαν σε μια βάση δεδομένων MySQL ώστε να διευκολυνθεί η διαχείριση και η αποδοτική αναζήτηση των εγγράφων. Σε αυτό το στάδιο, για κάθε δίπλωμα ευρεσιτεχνίας εξετάστηκαν τα βιβλιογραφικά δεδομένα (bibliographic data) σε ενοποιημένη μορφή, λαμβάνοντας υπόψη την πιο πρόσφατη έκδοση (π.χ. με βάση το πιο πρόσφατο kind code ή την ημερομηνία παραγωγής εγγράφου). Σκοπός ήταν να εντοπιστούν μόνο εκείνα τα έγγραφα που:

- είναι γραμμένα στην αγγλική γλώσσα,
- φέρουν τουλάχιστον έναν κύριο ταξινομητικό κωδικό (main IPC code),

- διαθέτουν και τα τρία βασικά πεδία: περίληψη (abstract), περιγραφή (description) και αξιώσεις (claims).



Εικόνα 2.3: : Διάγραμμα προϋποθέσεων για εξαγωγή υποσυνόλου CLEF-IP

Η λίστα με τα patent numbers και UCIDs που πληρούν τα παραπάνω κριτήρια αποθηκεύτηκε και χρησιμοποιήθηκε ως βάση για την περαιτέρω εξαγωγή πληροφορίας.

Στη συνέχεια, πραγματοποιήθηκε λεπτομερής ανάλυση (parsing) των αντίστοιχων XML αρχείων με χρήση της βιβλιοθήκης BeautifulSoup, με στόχο την εξαγωγή όλων των σχετικών πεδίων: κείμενα (τίτλος, abstract, description, claims), βιβλιογραφικά στοιχεία (ημερομηνίες, status, χώρα, kind code, γλώσσα), ταξινομήσεις (main, further, IPCR, ECLA), καθώς και τα ονόματα των εφευρετών (inventors) και των αιτούντων (applicants).

Ο κώδικας που υλοποιήθηκε περιλαμβάνει επαναληπτική αναζήτηση σε ολόκληρη τη δομή φακέλων της συλλογής και έλεγχο για την ύπαρξη των απαραίτητων πεδίων ανά δίπλωμα. Κατά την επεξεργασία, δόθηκε ιδιαίτερη έμφαση στην αποφυγή διπλοεγγραφών, την αποκατάσταση μη δομημένων δεδομένων, καθώς και στην κανονικοποίηση των ταξινομητικών κωδικών.

Το τελικό υποσύνολο που προέκυψε, αποθηκεύτηκε σε μορφή CSV, ενώ για την αποφυγή υπερφόρτωσης της μνήμης έγινε τμηματική αποθήκευση κάθε 500.000 εγγραφές. Επιπλέον, έχει διατεθεί προς το κοινό ένα αντιπροσωπευτικό δείγμα (sample subset) για άμεση χρήση και αξιολόγηση, ενώ ο πλήρης κώδικας εξαγωγής και καθαρισμού δημοσιεύεται ανοιχτά, επιτρέποντας την αναπαραγωγή των βημάτων από άλλους ερευνητές.

2.2 WPI

2.2.1 Περιγραφή της συλλογής WPI

Η συλλογή WPI αποτελεί ένα στατικό, και δημόσια διαθέσιμο σύνολο δεδομένων το οποίο δημιουργήθηκε για να ενθαρρύνει τη συγκρισιμότητα, την αναπαραγωγιμότητα και την επαναληπτικότητα σε πειράματα σχετικά με διπλώματα ευρεσιτεχνίας, παρέχοντας ένα σταθερό σώμα δεδομένων που δεν θα ενημερώνεται, ώστε οποιεσδήποτε αλλαγές στα αποτελέσματα των πειραμάτων να μπορούν να αποδίδονται με ασφάλεια σε μεθοδολογικές προόδους και όχι σε μεταβολές των δεδομένων.

Η συλλογή περιλαμβάνει περίπου 6.3 εκατομμύρια έγγραφα πατεντών, τα οποία καλύπτουν την χρονική περίοδο 2014-2015 που προέρχονται από διαφορετικές Αρχές ευρεσιτεχνίας όπως:

- Κίνα (CN)
- Ευρωπαϊκό Γραφείο Διπλωμάτων Ευρεσιτεχνίας (EP)
- Ιαπωνία (JP)
- Κορέα (KR)
- Ηνωμένες Πολιτείες (US)
- Παγκόσμιος Οργανισμός Διανοητικής Ιδιοκτησίας (WO)

Όπως και στην περίπτωση της CLEF-IP, έτσι και αυτή περιλαμβάνει το πλήρες κείμενο του διπλώματος ευρεσιτεχνίας αλλά και δομημένα μεταδεδομένα. Τα αρχεία της συλλογής είναι οργανωμένα κατά αρχή και κατά εβδομάδα. Δηλαδή στον φάκελο ”20140101” που υπάρχει μέσα στον φάκελο ”EP” υπάρχουν όλες οι πατέντες της πρώτης εβδομάδας του 2014.

Η συλλογή προσφέρει ένα σταθερό σύνολο δεδομένων για να μπορούν διάφορες ερευνητικές ομάδες να συγκρίνουν αποτελέσματα σε ακριβώς το ίδιο σώμα κειμένων, υποστηρίζοντας:

- Πειράματα ανάκτησης διπλωμάτων
- Εξαγωγή πληροφορίας (π.χ. εφευρέτης, δικαιούχος, ταξινόμηση)
- Ανάλυση παραπομπών και προγενέστερων τεχνικών
- Εργασίες επεξεργασίας φυσικής γλώσσας (κατηγοριοποίηση, ανίχνευση καινοτομίας κ.ά.)

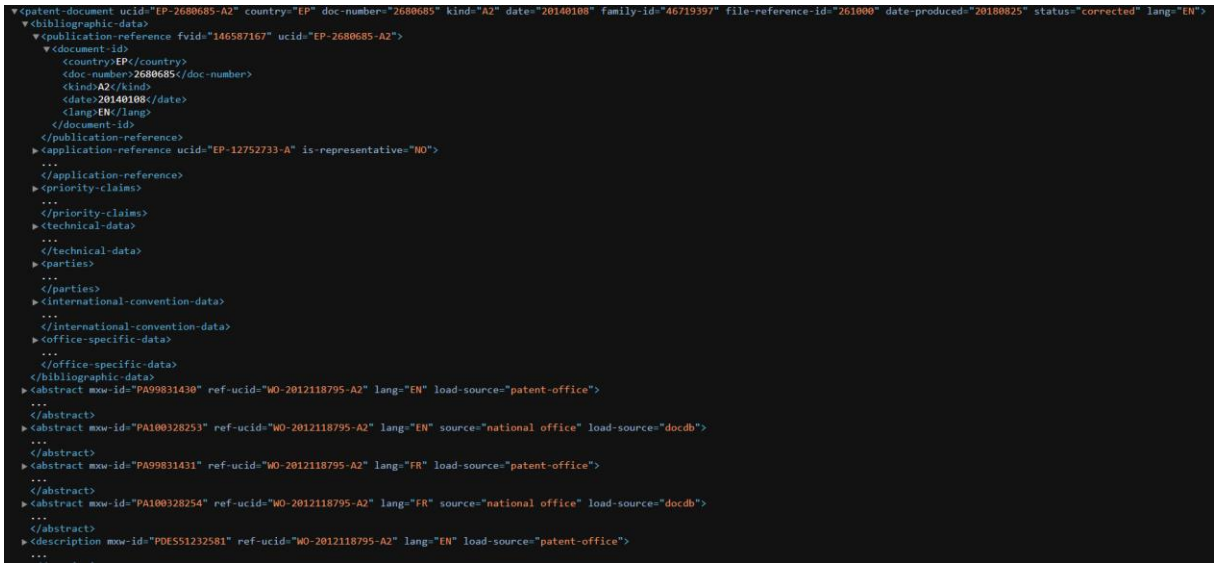
Σε αντίθεση με άλλες συλλογές που εστιάζουν σε έναν τομέα ή μία αρχή για πολλά έτη, η WPI είναι οριζόντια — καλύπτει όλους τους τομείς και τις αρχές κατά μία σύντομη, καλά καθορισμένη χρονική περίοδο

2.2.2 Δομή XML των αρχείων της WPI

Τα δεδομένα της συλλογής διατίθενται σύμφωνα με το πρότυπο ST.36 του WIPO σε μορφή XML. Κάθε XML αρχείο αντιστοιχεί σε μια συγκεκριμένη εκδοχή μιας πατέντας και περιλαμβάνει βιβλιογραφικά δεδομένα και το τεχνικό περιεχόμενο του εγγράφου.

Όπως και στην CLEF-IP, όλα τα στοιχεία κάθε XML της WPI περιέχονται στο εσωτερικό του στοιχείου <patent-document>, το οποίο περιλαμβάνει τα:

- <bibliographic-data> τα μεταδεδομένα (π,χ, γλώσσα, εκδοχή πατέντας, ημερομηνίες κατάθεσης και δημοσίευσης, ταξινομήσεις IPC κ.ά.)
- <abstract> η περίληψη του διπλώματος που μπορεί να υπάρχει σε διαφορετικές γλώσσες
- <description> το αναλυτικό τεχνικό κείμενο που περιγράφει την εφεύρεση, που μπορεί να υπάρχει σε διαφορετικές γλώσσες
- <claims> οι αξιώσεις που μπορεί να υπάρχουν σε διαφορετικές γλώσσες



Εικόνα 2.4: Δομή XML αρχείου συλλογής WPI

Όπως φαίνεται στην Εικόνα 2.4 ενός του στοιχείου <bibliographic-data> περικλείονται άλλα στοιχεία με τα πιο σημαντικά να είναι τα:

- <publication-reference> περιλαμβάνει την Αρχή ευρεσιτεχνίας, τον αριθμό πατέντας, την εκδοχή της πατέντας, την ημερομηνία έκδοσης και την γλώσσα
- <publication-reference> περιλαμβάνει αναφορές σε άλλες εκδοχές της πατέντας
- <priority-claims> περιλαμβάνει άλλες πατέντες από τις οποίες εξαρτάται το συγκεκριμένο έγγραφο
- <technical-data> περιλαμβάνει ταξινομήσεις IPC και CPC αλλά και τον τίτλο της πατέντας σε διαφορετικές γλώσσες
- <parties> περιλαμβάνει τους εφευρέτες και τους αιτούντες

Όπως και στην CLEF-IP, έτσι και στην WPI πολλές πατέντες υπάρχουν σε περισσότερες από μία εκδοχές, με διαφορετικά kind codes, κάτι που απαιτεί προσοχή κατά την ενοποίηση ή την επιλογή της πιο πρόσφατης/πληρέστερης έκδοσης ενώ παρόλο που τα περισσότερα πεδία υπάρχουν σε κάθε XML, η πληρότητα των στοιχείων διαφέρει

Κεφάλαιο 3ο: Οργάνωση dataset στο Hugging Face

Στο Κεφάλαιο αυτό θα αναλυθεί η πλατφόρμα Hugging Face έχει ως ένα από τα σημαντικότερα οικοσυστήματα εργαλείων και κοινοτήτων για την ανάπτυξη και διαμοιρασμό μοντέλων μηχανικής μάθησης, το κομμάτι του γράφου δηλαδή που φαίνεται στην Εικόνα 3.1

Εικόνα 3.1: Hugging Face στον Γράφο

3.1 Εισαγωγή στην πλατφόρμα Hugging Face

Αν και η αρχική της εστίαση ήταν στα γλωσσικά μοντέλα, σήμερα το Hugging Face είναι μια πλατφόρμα που φιλοξενεί 900 χιλιάδες μοντέλα, 200 χιλιάδες datasets και 300 χιλιάδες demo εφαρμογές (spaces) από διάφορους τομείς όπως computer vision, audio processing και reinforcement learning.

Μέσω του Hugging Face, οι ερευνητές και οι προγραμματιστές μπορούν να ανεβάζουν, να φιλοξενούν και να διαμοιράζονται:

- Προ-εκπαιδευμένα ή fine-tuned μοντέλα
- Datasets οποιουδήποτε τύπου (CSV, JSON, εικόνες κ.ά.)
- Notebooks, scripts, ακόμα και ολόκληρα Spaces (διαδραστικές εφαρμογές Machine Learning μέσω Gradio ή Streamlit)

Η κεντρική ιδέα πίσω από την πλατφόρμα είναι η υιοθέτηση open-source λογικής, με την προσέγγιση ότι η κοινότητα είναι ο κεντρικός πυλώνας του οικοσυστήματος. Το Hugging Face έχει μετατραπεί σε έναν χώρο συνεργασίας, όπου ερευνητές, ακαδημαϊκοί, και μηχανικοί εταιρειών μοιράζονται πρότυπα, βέλτιστες πρακτικές, συλλογές δεδομένων και κώδικα. Η διάθεση open-source εργαλείων (π.χ. Transformers, Tokenizers) έχει διευκολύνει σημαντικά την υιοθέτηση σύγχρονων τεχνικών στην ευρύτερη ερευνητική και βιομηχανική κοινότητα. Το Hugging Face διακρίνεται όχι μόνο για το τεχνικό του υπόβαθρο αλλά και για τη φιλοσοφία διαφάνειας και προσβασιμότητας που προάγει [6].

Στην καρδιά του οικοσυστήματος Hugging Face βρίσκονται τα αποθετήρια **(**repositories**)**, τα οποία λειτουργούν ως κόμβοι οργάνωσης και διάθεσης δεδομένων, μοντέλων και εφαρμογών. Τα αποθετήρια φιλοξενούνται στο Hugging Face ως Git repositories, που σημαίνει ότι η συνεργασία και το version control αποτελούν θεμελιώδη στοιχεία της πλατφόρμας. Η πλατφόρμα παρέχει ενιαία υποστήριξη για τρεις βασικούς τύπους αποθετηρίων: models, datasets, και spaces. [7]

Dataset Repositories: Τα dataset repositories επιτρέπουν την οργάνωση, διαμοίραση και φόρτωση συλλογών δεδομένων με μεταδεδομένα, τεκμηρίωση και custom loading scripts. Ο εκάστοτε χρήστης της πλατφόρμας μπορεί να ανεβάσει CSVs, JSON, Parquet ή ακόμη και αρχεία ήχου/εικόνας συνοδεύοντας τα με αρχείο τεκμηρίωσης README. Ακόμα υπάρχει η δυνατότητα προσθήκης tags (π.χ. language:en, task:classification) για ευκολότερη αναζήτηση και η ενσωμάτωση custom data loaders σε Python χρησιμοποιώντας την βιβλιοθήκη datasets. [8]

Model Repositories: Τα model repositories φιλοξενούν προ-εκπαιδευμένα ή fine-tuned Machine Learning μοντέλα. Το Hugging Face προσφέρει model cards τα οποία περιγράφουν την αρχιτεκτονική, τα δεδομένα εκπαίδευσης κα του περιορισμούς των μοντέλων. Επίσης υπάρχει η δυνατότητα ενσωμάτωσης σε API, επιτρέποντας άμεση δοκιμή μοντέλων μέσω browser ή RESTful endpoints. [9]

Space Repositories: Τα space repositories επιτρέπουν την ανάπτυξη διαδραστικών εφαρμογών που «τρέχουν» πάνω στην υποδομή της πλατφόρμας. Με χρήση εργαλείων όπως Gradio ή Streamlit, μπορούν να δημιουργηθούν live demos εφαρμογών που μπορούν να χρησιμοποιηθούν για πολλές διεργασίες όπως image, video και text generation αλλά και οπτικοποιήσεις. [10]

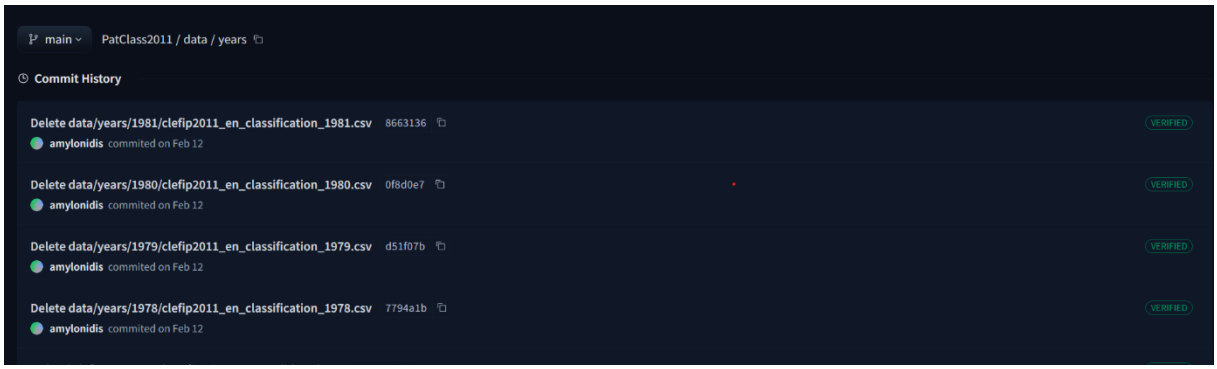
3.2 Δημιουργία και διαχείριση αποθετηρίου

Στο πλαίσιο της παρούσας εργασίας η πλατφόρμα Hugging Face αξιοποιήθηκε ώστε να αποθηκευτεί ένα υποσύνολο δεδομένων της συλλογής CLEF-IP στοχευμένο στην ταξινόμηση, το οποίο θα μπορεί να αποτελέσει αναφορά για ερευνητές στο πεδίο της ανάλυσης πατεντών και συγκεκριμένα της ταξινόμησης πατεντών, ώστε να μπορούν να επαναλάβουν και να συγκρίνουν μεθόδους στα ίδια δεδομένα. Περισσότερες πληροφορίες για τον τρόπο που παράχθηκε το υποσύνολο αναφέρονται στο Κεφάλαιο 2. Για τον σκοπό αυτό δημιουργήθηκε ένα αποθετήριο δεδομένων (dataset repository).

Η δημιουργία του repository έγινε από το web interface της πλατφόρμας, επιλέγοντας το *New Dataset* . Μετά από αυτή την ενέργεια είναι απαραίτητο να συμπληρωθούν κάποια απαραίτητα στοιχεία για το dataset. Επιλέχθηκε το repository να είναι public και το license που επιλέχθηκε είναι το MIT. Το ότι το repository είναι public σημαίνει ότι οποιοσδήποτε χρήστης, ακόμα και αν δεν έχει λογαριασμό στο Hugging Face μπορεί να δει και να κατεβάσει τα αρχεία.

Το MIT είναι από τις πιο ανοιχτές και ανεκτικές άδειες, γι' αυτό και τη χρησιμοποιούν πολλά έργα ανοιχτού κώδικα, ειδικά στο Hugging Face και στο GitHub. Το συγκεκριμένο license επιτρέπει στον κάθε χρήστη να χρησιμοποιήσει, αντιγράψει, τροποποιήσει, συγχωνεύσει, δημοσιεύσει και να διανείμει το εκάστοτε έργο είτε για προσωπική είτε για εμπορική χρήση. Ο χρήστης υποχρεούται ωστόσο να συμπεριλάβει πάντα στο project του την πρωτότυπη άδεια MIT και την αναφορά στον αρχικό δημιουργό [11].

Το ανέβασμα των αρχείων για το dataset έγινε μέσα από το UI της πλατφόρμας. Καθώς τα dataset repositories αποτελούν Git repositories υπάρχει η δυνατότητα το ανέβασμα να γίνει χρησιμοποιώντας git operations, στην συγκεκριμένη περίπτωση το push. Αυτό δεν θεωρήθηκε σκόπιμο αφού πολλά από τα αρχεία προς ανέβασμα είχαν μεγάλο μέγεθος (πάνω από 1 GB) , για αυτό και προτιμήθηκε το UI το οποίο είναι πιο διαδραστικό. Για την διαχείριση του dataset όμως έγινε χρήση των λειτουργιών του git καθώς προσφέρουν στρατηγικά πλεονεκτήματα στην διαχείριση των δεδομένων. Αρχικά, η ανίχνευση των αλλαγών ήταν πολύ απλή, αφού κάθε αλλαγή στο dataset (π.χ. διόρθωση, προσθήκη νέων εγγραφών) καταγράφεται με **timestamp**, **author** και **commit message**, επιτρέποντας την πλήρη παρακολούθηση της ιστορίας του dataset όπως φαίνεται και στο commit history στην Εικόνα 3.2



Εικόνα 3.2: Commit History

Επιλέγοντας οποιοδήποτε από αυτά τα commits είναι ορατό σε ποια σημεία έχουν γίνει αλλαγές γραμμή προς γραμμή σε κάθε αρχείο κατά αυτό το commit. Επίσης σε περίπτωση λανθασμένων αλλαγών η διαγραφών η αποκατάσταση των δεδομένων έγινε με ευκολία καθώς οι λειτουργίες του git επιτρέπουν την επιστροφή σε προηγούμενες εκδόσεις με ασφάλεια και χωρίς απώλεια δεδομένων. Ακριβώς επειδή στα repositories είναι αποθηκευμένες όλες οι εκδόσεις των αρχείων που έχουν ανέβει, άρα στην ουσία και διαφορετικά στιγμιότυπα του dataset, υπάρχει η δυνατότητα να εκτελέσει κανείς πειράματα σε διαφορετικές εκδόσεις του dataset αβίαστα. Τέλος αν αυτό είναι επιθυμητό, πολλοί χρήστες μπορούν να δουλεύουν σε διαφορετικές εκδόσεις του dataset ταυτόχρονα χρησιμοποιώντας διαφορετικά branches του repository.

Το συνολικό μέγεθος του dataset είναι 67.4 GB. Ένα σημαντικό πλεονέκτημα του Hugging Face είναι ότι στην περίπτωση του δωρεάν λογαριασμού, για τα public repositories ο αποθηκευτικός χώρος είναι δωρεάν, ενώ για τα private είναι επί πληρωμή όταν ξεπεράσει τα 100 GB. Στην περίπτωση που δημιουργηθεί PRO λογαριασμός, ο οποίος είναι επί πληρωμή, ο διαθέσιμος αποθηκευτικός χώρος είναι απεριόριστος για τα public repositories ενώ για τα private είναι δωρεάν το πρώτο TB και στη συνέχεια το κόστος ανεβαίνει αναλογικά με την χρήση (pay-as-you-go). Στην περίπτωση του δωρεάν λογαριασμού για τα public repositories δεν αναφέρεται κάποιο συγκεκριμένο όριο στον αποθηκευτικό χώρο που μπορεί να χρησιμοποιηθεί, αναφέρεται όμως ότι υπάρχουν διαδικασίες οι οποίες ώστε να μην γίνεται κατάχρηση του δωρεάν ελεύθερου χώρου χωρίς να είναι σαφές ποιες είναι αυτές. Επίσης υπάρχει η σύσταση στους χρήστες της πλατφόρμας να προσπαθούν έτσι ώστε τα δεδομένα που ανεβάζουν να είναι όσο το δυνατόν πιο χρήσιμα στην κοινότητα και να μην φιλοξενούνται δεδομένα άσκοπα [12]. Ένα σαφές συμπέρασμα σχετικά με τον αποθηκευτικό χώρο στο Hugging Face, είναι ότι η πλατφόρμα προσφέρει με γενναιοδωρία πόρους και ενδείκνυται ακόμα και για μεγάλο όγκο δεδομένων. Για την παρούσα εργασία το κόστος ήταν μηδενικό, ενώ πραγματοποιήθηκαν δοκιμαστικά uploads μεγάλων αρχείων, 25 GB το μεγαλύτερο, χωρίς κανένα πρόβλημα.

3.3 Τεκμηρίωση και περιγραφή του CLEF-IP-2011_EN_All_MainClass dataset

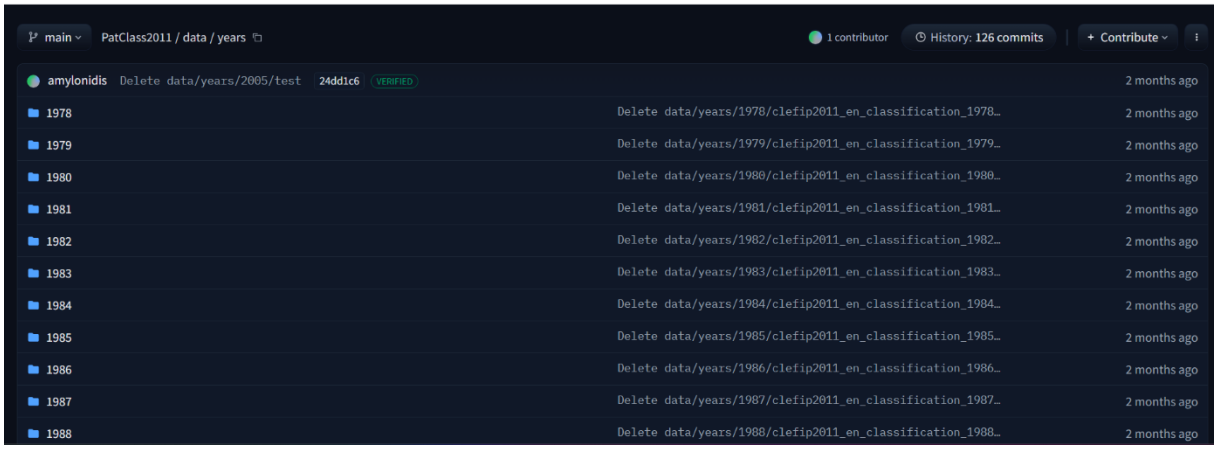
Το dataset αποτελείται από 28 αρχεία CSV τα οποία περιέχουν τα οποία περιέχουν τα δεδομένα του υποσυνόλου που περιεγράφηκε στο Κεφάλαιο 2.1.4 και στο σύνολο τους αποτελούνται από 718,834 εγγραφές .

Κάθε CSV αρχείο του dataset έχει την ίδια δομή, η οποία είναι οι παρακάτω 19 στήλες:

- **ucid:** Το μοναδικό αναγνωριστικό του εγγράφου το οποίο συγκροτείται από την χώρα, το doc-number και το ειδος της πατέντας π.χ. WO-1978000001-A1
- **doc_number:** Ο μοναδικός αριθμός αναγνώρισης μιας πατέντας, ο οποίος αποδίδεται από την αντίστοιχη αρμόδια αρχή πνευματικής ιδιοκτησίας πχ 1978000001

- **country:** Ο κωδικός της χώρας ή του οργανισμού που είναι αρμόδιος για την έκδοση της πατέντας π.χ. EP (European Patent Office), US (United States Patent and Trademark Office) ή WO (WIPO - World Intellectual Property Organization)
- **kind: O** αλφαριθμητικός κωδικός που καθορίζει το είδος και το στάδιο της πατέντας, δηλαδή αν πρόκειται για αρχική δημοσίευση αίτησης, τελική χορήγηση, ή διορθωμένη έκδοση π.χ. B1
- **lang:** Η γλώσσα στην οποία έχει συνταχθεί ή δημοσιευθεί η πατέντα π.χ. EN (Αγγλικά), FR (Γαλλικά), DE (Γερμανικά)
- **date:** Η ημερομηνία επίσημης δημοσίευσης μιας πατέντας από τον αντίστοιχο οργανισμό πνευματικής ιδιοκτησίας (π.χ. EPO, USPTO, WIPO). Η ημερομηνία αυτή σηματοδοτεί τη στιγμή κατά την οποία το έγγραφο καθίσταται δημόσια προσβάσιμο. Η μορφή της είναι YYYYMMDD π.χ. 19991005
- **application_date:** Η ημερομηνία κατάθεσης της πατέντας, δηλαδή τη στιγμή κατά την οποία ο αιτών υπέβαλε για πρώτη φορά την αίτηση ευρεσιτεχνίας στον αρμόδιο οργανισμό πνευματικής ιδιοκτησίας (π.χ. EPO, USPTO, WIPO). Η μορφή της είναι YYYY π.χ. 1978
- **date_produced:** Η ημερομηνία κατά την οποία εισάχθηκε η πατέντα στο dataset. Η μορφή της είναι YYYYMMDD π.χ. 20090919
- **status:** Η τρέχουσα νομική ή διαδικαστική κατάσταση της πατέντας π.χ., new
- **main_code:** Ο κύριος κωδικός ταξινόμησης που έχει αποδοθεί σε μια πατέντα βάσει του συστήματος IPC (International Patent Classification). Ο main_code αναπαριστά τη βασική τεχνολογική κατηγορία στην οποία εντάσσεται η πατέντα π.χ. F16L13
- **further_codes:** Ένας ή περισσότεροι επιπλέον κωδικοί ταξινόμησης IPC, οι οποίοι αποδίδονται σε μια πατέντα προκειμένου να περιγράψουν δευτερεύουσες τεχνολογικές πτυχές της εφεύρεσης, οι οποίες δεν καλύπτονται πλήρως από τον main_code. Οι further_codes επιτρέπουν μια πιο λεπτομερή και κατηγοριοποίηση, αποτυπώνοντας όλες τις επιμέρους τεχνικές λύσεις ή εφαρμογές που σχετίζονται με την εφεύρεση π.χ. F16L23, F16L47, B23P11
- ipcr_**codes:** Ένας ή περισσότεροι κωδικοί ταξινόμησης IPC σε επίπεδο έκδοσης IPCR. Η IPCR αποτελεί τη μεταρρυθμισμένη μορφή του συστήματος IPC, που εφαρμόζεται από το 2006 και μετά, παρέχοντας πιο λεπτομερείς, ενημερωμένες και πολυεπίπεδες ταξινομήσεις π.χ. B23P11, F16L13, F16L23, F16L47
- **ecla_codes:** Ένας ή περισσότεροι κωδικοί ταξινόμησης ECLA, δηλαδή ένα σύστημα ταξινόμησης που χρησιμοποιούνταν από EPO για τον εμπλουτισμό και την επέκταση της IPC ταξινόμησης με περισσότερες υποκατηγορίες και λεπτομέρειες π.χ. B23P 11/02B, F16L 13/00C, F16L 23/024, F16L 47/22
- **title:** Ο τίτλος της πατέντας, όπως έχει αποδοθεί από τον καταθέτη ή το αρμόδιο γραφείο πατεντών κατά τη διαδικασία υποβολής ή δημοσίευσης του εγγράφου. Ο τίτλος συνοψίζει τη βασική τεχνική εφεύρεση ή το αντικείμενο της αίτησης, συχνά με συνοπτικό, τεχνικό και περιγραφικό τρόπο π.χ. “PIPES AND COUPLINGS AND METHOD OF COUPLING PIPES”
- **abstract:** Η περίληψη της πατέντας, η οποία συνοψίζει με τεχνικό και περιεκτικό τρόπο την εφεύρεση και τη λειτουργικότητά της. Αποτελεί ουσιώδες στοιχείο της πατέντας και έχει ως σκοπό την εύκολη κατανόηση του αντικειμένου της εφεύρεσης από μη εξειδικευμένο κοινό
- **description:** Η περιγραφή περιλαμβάνει τη λεπτομερή τεχνική παρουσίαση της εφεύρεσης, εξηγώντας τον σκοπό, τα πλεονεκτήματα, και τον τρόπο υλοποίησης της.
- **claims:** Το νομικά δεσμευτικό τμήμα μιας πατέντας (αξιώσεις), το οποίο καθορίζει το εύρος της προστασίας που παρέχεται από την εφεύρεση. Οι αξιώσεις περιγράφουν με τεχνικό τρόπο τα στοιχεία που θεωρούνται καινοτόμα, διαχωρίζοντας την εφεύρεση από την προηγούμενη τεχνολογία (prior art)
- **applicants:** Τα φυσικά ή νομικά πρόσωπα που υπέβαλαν την αίτηση ευρεσιτεχνίας, δηλαδή εκείνους που ζητούν νομική προστασία για την εφεύρεση. Σε αντίθεση με τον εφευρέτη (inventor), ο αιτών **(**applicant**)** μπορεί να είναι ο ίδιος ο εφευρέτης, αλλά συχνότερα είναι εταιρικός φορέας**,** ερευνητικό ίδρυμα ή ιδιώτης χρηματοδότης, που κατέχει τα νομικά δικαιώματα π.χ. “INT WATER SAVING SYST INC; INTERNATIONAL WATER SAVING SYSTEMS INC”
- **inventors:** Τα φυσικά πρόσωπα που συνέλαβαν την ιδέα και συνέβαλαν ουσιαστικά στην ανάπτυξη της εφεύρεσης π.χ. “ALBERTASSI J H; HEINZE W O”

Τα παραγόμενα CSV που ανέβηκαν στο Hugging Face είναι οργανωμένα ανά έτος. Κάθε CSV αντιστοιχεί σε ένα έτος, από το 1978 μέχρι και το 2005 και κάθε CSV βρίσκεται σε φάκελο με όνομα το εκάστοτε έτος όπως φαίνεται και στην Εικόνα 3.3.



Εικόνα 3.3: Δομή Dataset

Κάθε φάκελος που φαίνεται στην Εικόνα 3.3 περιέχει ένα CSV αρχείο. Αντί για ένα μεγάλο ενιαίο αρχείο, η χρήση φακέλων και αρχείων ανά χρονολογία του dataset έχει σημαντικές πρακτικές προεκτάσεις. Αρχικά, επιτρέπει την αποδοτικότερη φόρτωση των δεδομένων αφού μπορεί να γίνει επιλεκτική φόρτωση μόνο των δεδομένων που απαιτούνται κάθε φορά. Αυτό έχει σαν αποτέλεσμα την μείωση του απαιτούμενου χρόνου και της μνήμης RAM που χρειάζεται για αναλύσεις. Ένα ακόμα πλεονέκτημα αυτής της δομής είναι ότι έχει σαν αποτέλεσμα μια πιο φυσική μοντελοποίηση των χρονικών δεδομένων. Ο διαχωρισμός ανά έτος υποστηρίζει άμεσα χρονολογικές αναλύσεις οι οποίες είναι πολύ σημαντικές καθώς η χρονική διάσταση είναι κρίσιμη στα δεδομένα πατεντών, αφού η τεχνολογική εξέλιξη συχνά εκτιμάται ανά έτος ή δεκαετία. Ακόμα, η προσθήκη νέων δεδομένων (π.χ. νέα έτη ή αναθεωρήσεις) γίνεται απλά με την δημιουργία ενός νέου αρχείου CSV, χωρίς να απαιτείται επαναφόρτωση ή τροποποίηση του συνόλου του dataset. Αυτό επιτρέπει στο repository να είναι εύκολα επεκτάσιμο. Τέλος, η οργάνωση αυτή κάνει πιο εύκολη την παραλληλοποίηση εργασιών και πιο συγκεκριμένα την παράλληλη επεξεργασία σε κατανεμημένα συστήματα ή multi threaded περιβάλλοντα.

Η τεκμηρίωση αποτελεί κρίσιμο συστατικό στοιχείο ενός ποιοτικού dataset repository, καθώς εξασφαλίζει ότι οι χρήστες κατανοούν τη φύση και τη δομή των δεδομένων και διευκολύνει ενδεχόμενη συνεισφορά τους στο project. Επίσης, κάνει εφικτή την αναπαραγωγή των ερευνητικών αποτελεσμάτων και ενισχύει την υπευθυνότητα και η διαφάνεια ως προς τη χρήση των δεδομένων. Στο Hugging Face η τεκμηρίωση πραγματοποιείται επίσημα δημιουργώντας ένα Dataset Card [13]. Ένα Dataset Card είναι ένα ειδικό αρχείο τεκμηρίωσης που συνοδεύει κάθε dataset και παρέχει ενδεικτικά πληροφορίες σχετικά με:

- Περιγραφή του dataset (Περίληψη dataset, Υποστηριζόμενες Διεργασίες, Γλώσσες)
- Δομή του dataset (Στιγμιότυπα των Δεδομένων, Πεδία των Δεδομένων, Υποσύνολα των Δεδομένων, Φόρτωση των Δεδομένων)
- Δημιουργία του dataset (Επιμέλεια, Προέλευση Δεδομένων, Παρατηρήσεις, Προσωπικά και Ευαίσθητα Δεδομένα)

- Θέματα Χρήσης των Δεδομένων (Κοινωνικός Αντίκτυπος, Θέματα Ηθικής)
- Επιπρόσθετες Πληροφορίες (Επιμελητές, Άδεια Χρήσης, Τρόπος Αναφοράς)

Τα Dataset Cards υλοποιούνται σε μορφή **Markdown** (README.md αρχεία) και το Huggung Face τα εμφανίζει στην κεντρική σελίδα κάθε dataset.

Ένα ακόμα σημαντικό δομικό στοιχείο των Dataset Cards αποτελούν τα metadata τα οποία περιγράφουν πληροφορίες όπως την άδεια χρήσης, την γλώσσα, το μέγεθος του dataset και κατηγορίες που ανήκει. Αυτά τα στοιχεία χρησιμοποιούνται από το Hugging Face για σκοπούς ευρετηρίασης, δηλαδή οι χρήστες μπορούν να αναζητούν datasets στην πλατφόρμα χρησιμοποιώντας φίλτρα με βάση αυτά τα στοιχεία. Όταν δηλαδή ένας χρήστης αναζητά στο Hugging Face datasets με μέγεθος μέχρι 30 GB, το σύστημα του εμφανίζει τα datasets που έχουν καταχωρημένο σαν metadata στο Dataset Card τους το συγκεκριμένο στοιχείο. Είναι λοιπόν σημαντικό για datasets που απευθύνονται σε μεγάλο αριθμό χρηστών να έχουν σωστά καθορισμένα τα metadata τους.

3.4 Φόρτωση dataset

Η διαδικασία φόρτωσης ενός dataset αποτελεί θεμελιώδες στάδιο σε κάθε έργο που βασίζεται στην ανάλυση δεδομένων. Ιδίως στην περίπτωση των δεδομένων πατεντών, τα οποία είναι συνήθως μεγάλα σε όγκο και πολυδιάστατα η ορθή φόρτωσή τους αποκτά ακόμη μεγαλύτερη σημασία. Η επιτυχής φόρτωση του dataset διασφαλίζει:

- Ομοιογένεια και ακρίβεια στη μορφοποίηση των δεδομένων, πράγμα απαραίτητο για μεταγενέστερη επεξεργασία και ανάλυση
- Βελτιστοποίηση πόρων (μνήμη, χρόνος) με τη χρήση κατάλληλων τεχνικών
- Αποσύνδεση της ανάλυσης από τη φυσική τοποθεσία των δεδομένων

Ιδιαίτερα στα projects που σχετίζονται με πατέντες, όπου οι αναλυτικές διαδικασίες περιλαμβάνουν πολυεπίπεδη επεξεργασία (π.χ. εξαγωγή ταξινομήσεων, χρονικές σειρές, ανάλυση κειμένων κ.ά.), η σωστή φόρτωση αποτελεί τον ακρογωνιαίο λίθο για την αξιόπιστη εξαγωγή συμπερασμάτων. Είναι ξεκάθαρο ότι η φόρτωση του dataset δεν είναι απλώς ένα τεχνικό βήμα, αλλά διασφαλίζει την ποιότητα, την ακεραιότητα και την χρηστικότητας των δεδομένων.

Στο πλαίσιο της παρούσας εργασίας αναπτύχθηκε ένα custom loading script προσαρμοσμένο στις ανάγκες του dataset, το οποίο χρησιμοποιεί τον μηχανισμό φόρτωσης datasets του Hugging Face και θα αναλυθεί παρακάτω.

3.4.1 Το loading script του Hugging Face

Το Hugging Face προσφέρει την Python βιβλιοθήκη datasets η οποία χρησιμοποιείται για την φόρτωση dataset repositories που φιλοξενούνται στην πλατφόρμα με την χρήση μιας γραμμής κώδικα και για την αποδοτική προ-επεξεργασία των δεδομένων σε πολλές μορφές όπως CSV, JSON, PNG, JPEG WAV και Parquet[14]. Ένα παράδειγμα φόρτωσης dataset φαίνεται στην Εικόνα 3.4

```
>>> from datasets import load_dataset
>>> dataset = load_dataset("lhoestq/demo1")
```

Εικόνα 3.4: Παράδειγμα φόρτωσης dataset

Η πρώτη εντολή φορτώνει την συνάρτηση load_dataset από την βιβλιοθήκη datasets ώστε να μπορεί να χρησιμοποιηθεί. Η δεύτερη εντολή χρησιμοποιεί τη load_dataset για να κατεβάσει και να φορτώσει το dataset που βρίσκεται στο αποθετήριο **lhoestq/demo1** στο Hugging Face. Συγκεκριμένα, **lhoestq** είναι το όνομα του χρήστη ή οργανισμού στο Hugging Face και **demo1** είναι το όνομα του dataset μέσα στο χώρο αυτού του χρήστη. Η load_dataset θα ανακτήσει τα αρχεία του dataset (όπως CSV, JSON κ.ά.), θα τα αποθηκεύσει τοπικά (caching) και θα τα μετατρέψει σε ένα αντικείμενο τύπου DatasetDict ή Dataset, το οποίο μπορεί πολύ εύκολα να μετατραπεί σε ένα pandas dataframe[15]. Η μεταβλητή dataset είναι συνήθως ένα DatasetDict (αν το dataset έχει διαφορετικά splits, π.χ. train/test/validation) ή ένα απλό Dataset αντικείμενο, αν υπάρχει μόνο ένα split. Αν δεν έχει οριστεί κάποιο split στο dataset repository τότε κατά την φόρτωση του όλα τα δεδομένα θα μπουν στο train split.

Υπάρχει η δυνατότητα κατά την φόρτωση να οριστεί η έκδοση του dataset που είναι επιθυμητή όπως φαίνεται στη Εικόνα 3.5

```
>>> dataset = load_dataset(
...   "lhoestq/custom_squad",
...   revision="main"  # tag name, or branch name, or commit hash
... )
```

Εικόνα 3.5: Φόρτωση dataset με καθορισμό έκδοσης

Με την παράμετρο revision ορίζεται να φορτωθεί η έκδοση του dataset που βρίσκεται στο main branch. Αυτή η μέθοδος φόρτωσης είναι ιδιαίτερα χρήσιμη όταν υπάρχει η ανάγκη να εκτελεστεί το ίδιο κομμάτι σε διαφορετικές εκδόσεις του dataset για την εξαγωγή αναλύσεων ή συγκριτικών αποτελεσμάτων.

Μια ακόμα χρήσιμη δυνατότητα είναι να οριστεί συγκεκριμένο αρχείο από όλο το dataset προς φόρτωση ή και ολόκληρος φάκελος όπως φαίνεται στην Εικόνα 3.6

```
>>> from datasets import load_dataset

# load files that match the grep pattern
>>> c4_subset = load_dataset("allenai/c4", data_files="en/c4-train.0000*-of-01024.json.gz")

# load dataset from the en directory on the Hub
>>> c4_subset = load_dataset("allenai/c4", data_dir="en")
```

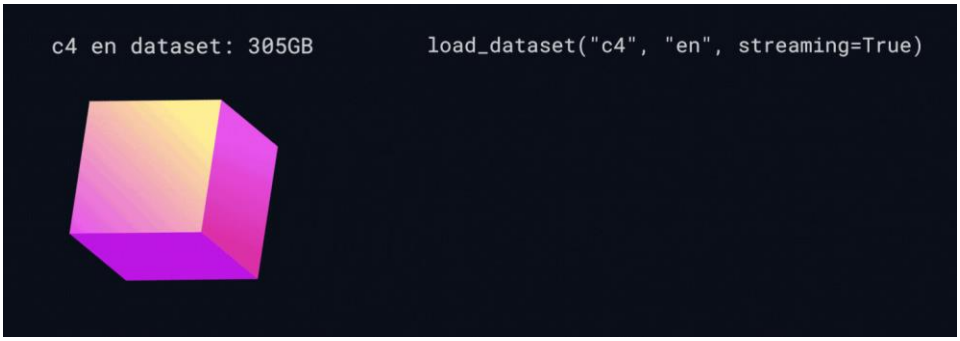
Εικόνα 3.6: Φόρτωση συγκεκριμένου αρχείου ή φακέλου

Στην συγκεκριμένη περίπτωση με την παράμετρο data_files ορίζεται να φορτωθεί από το dataset μόνο το αρχείο en/c4-train.0000*-of-01024.json.gz αλλά η συγκεκριμένη παράμετρος μπορεί να δεχθεί και λίστα αρχείων προς φόρτωση. Η παράμετρος data_dir χρησιμοποιείται για να φορτωθεί συγκεκριμένος φάκελος μέσα από ένα dataset, στην συγκεκριμένη περίπτωση θα φορτωθούν όλα τα αρχεία που υπάρχουν στον φάκελο en.

Ένα σημαντικό χαρακτηριστικό της συνάρτησης load_dataset είναι ότι το κατέβασμα του dataset θα εκτελεστεί μόνο την πρώτη φορά της εκτέλεσης. Την πρώτη φορά που θα γίνει download το dataset θα αποθηκευτεί στην μνήμη cache της huggingface_hub, η οποία είναι η βιβλιοθήκη για την αλληλεπίδραση με το Hugging Face[16]. Η huggingface_hub αξιοποιεί τον δίσκο του μηχανήματος στον οποίο είναι εγκατεστημένη, σαν μνήμη cache ώστε να αποθηκεύει τοπικά στην διαδρομή

~/.cache/huggingface/hub τα dataset που κατεβάζει πρώτη φορά[17]. Έτσι όταν θα φορτωθεί κάποιο dataset από αυτά μελλοντικά, δεν θα το ξανά-κατεβάσει από το Hugging Face, αλλά θα φορτωθεί από τον τοπικό φάκελο χωρίς να χρειάζεται καν σύνδεση στο Διαδίκτυο.

Πολλές φορές υπάρχει η ανάγκη να γίνει η εξερεύνηση των δεδομένων, ίσως και κάποιες δοκιμαστικές διεργασίες, ενός πολύ μεγάλου dataset το οποίο ξεπερνά το μέγεθος της μνήμης του χρήστη που θέλει να προβεί σε αυτές τις ενέργειες (π.χ. 10 TB) και χωρίς τεράστια αναμονή. Αυτή την ανάγκη καλύπτει το Stream [18].



Εικόνα 3.7: load_dataset με Stream

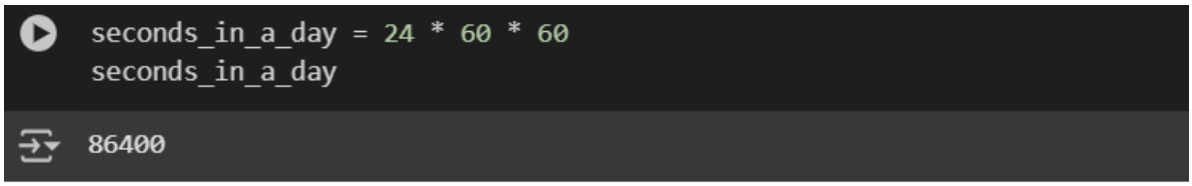
Όπως φαίνεται και στην Εικόνα 3.7, χρησιμοποιώντας την παράμετρο streaming=True τα δεδομένα του dataset φορτώνονται καθώς γίνεται η προσπέλαση τους (on-the-fly) από τον χρήστη δυναμικά και όχι όλα μαζί εξ αρχής. Με την χρήση της τεχνικής αυτής εξαλείφονται οι μεγάλοι χρόνοι αναμονής και οι υψηλές απαιτήσεις για μνήμη RAM, σε περιπτώσεις που δεν είναι επιτακτική ανάγκη η φόρτωση όλων των δεδομένων μαζί. Παράδειγμα χρήσης αποτελεί η εκπαίδευση ενός μοντέλου μηχανικής μάθησης, κατά την οποία τα δεδομένα μπορούν να επεξεργάζονται από την GPU σε μικρές παρτίδες χωρίς να χρειάζεται να φορτωθούν όλα στην RAM.

Είναι σαφές πως το Hugging Face παρέχει έναν εύχρηστο και αποδοτικό τρόπο διαχείρισης μεγάλων συνόλων δεδομένων με την χρήση της ευέλικτης μεθόδου load_dataset. Πολλές φορές όμως δεν αρκεί μόνο η χρήση αυτής της μεθόδου καθώς κάθε dataset είναι διαφορετικό, έχει τις δικές του ιδιομορφίες και χρήζει ειδικής μεταχείρισης στο στάδιο της φόρτωσης των δεδομένων. Για τους λόγους αυτούς στο πλαίσιο της παρούσας εργασίας, αναπτύχθηκε custom script για την φόρτωση των δεδομένων.

3.4.2 Το περιβάλλον εργασίας Google Colab

Η ανάπτυξη και η εκτέλεση του custom loading script, αλλά και ο κώδικας για την εξαγωγή αναλυτικών δεδομένων που θα αναλυθεί στο Κεφάλαιο 4, έγινε στο περιβάλλον εργασίας Google Colab. Το Google Colaboratory ή Google Colab είναι μια διαδικτυακή πλατφόρμα που παρέχεται από τη Google και επιτρέπει την εκτέλεση Python κώδικα μέσα από το περιβάλλον του προγράμματος περιήγησης (browser), χωρίς την ανάγκη τοπικής εγκατάστασης[19]. Στο Google Colab υπάρχει η έννοια του σημειωματάριου. Κάθε σημειωματάριο στο Colab είναι μια φιλοξενούμενη υπηρεσία σημειωματάριου Jupyter[20] που δεν απαιτεί κάποια εγκατάσταση για να εκτελεστεί. Πρόκειται για ένα διαμοιραζόμενο έγγραφο που συνδυάζει κώδικα, απλό κείμενο, HTML, δεδομένα ακόμα και οπτικοποιήσεις 3D μοντέλων. Όταν ένας χρήστης δημιουργεί ένα σημειωματάριο Colab αυτό αποθηκεύεται στον λογαριασμό Google Drive[21] του χρήστη. Έτσι ένας χρήστης μπορεί εύκολα να διαμοιράσει τα σημειωματάρια του σε άλλους χρήστες επιτρέποντας τους να τα σχολιάσουν είτε να κάνουν αλλαγές σε αυτά, ανάλογα με τα δικαιώματα που χορηγεί σε κάθε χρήστη.

Η δομή των σημειωματάριων είναι βασισμένη στα κελιά. Κάθε κελί περιέχει μία η περισσότερες γραμμές κώδικα και είναι εκτελέσιμο οποιαδήποτε στιγμή επιλέξει ο χρήστης κάνοντας κλικ στο κουμπί της αναπαραγωγής που βρίσκεται αριστερά πάνω στο κελί όπως φαίνεται και στην Εικόνα 3.8

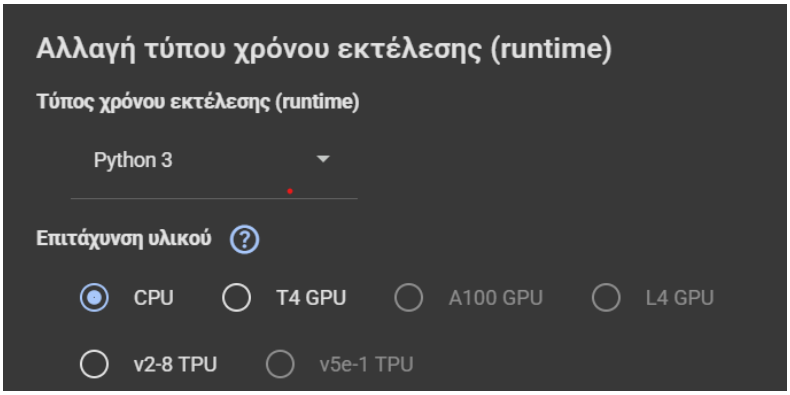


Εικόνα 3.8: Κελί κώδικα στο Google Colab

Η πρώτη γραμμή αποθηκεύει στην μεταβλητή seconds_in_a_day το αποτέλεσμα ενός υπολογισμού και η δεύτερη προβάλλει αυτό το αποτέλεσμα. Οι μεταβλητές που ορίζονται σε ένα κελί μπορούν να χρησιμοποιηθούν αργότερα σε άλλα κελιά.

Πολύ χρήσιμο χαρακτηριστικό αποτελεί το γεγονός ότι το Colab δεν απαιτεί καμία τοπική εγκατάσταση λογισμικού ή βιβλιοθηκών στον υπολογιστή του χρήστη. Όλα τρέχουν στην cloud υποδομή της Google και ο χρήστης χρειάζεται μόνο έναν browser και έναν λογαριασμό Google. Κατά αυτό τον τρόπο δεν υπάρχει καμία εξάρτηση από το λειτουργικό του χρήστη και υπάρχει αμεσότητα στην ανάπτυξη του κώδικα αφού ο χρήστης αποδεσμεύεται από το στήσιμο του περιβάλλοντος με εγκαταστάσεις πακέτων και βιβλιοθηκών. Η εγκατάσταση βιβλιοθηκών (π.χ. scikit-learn, pandas) γίνεται on-demand μέσα στο notebook με απλές εντολές (π.χ. !pip install)

Ένα από τα πιο σημαντικά πλεονεκτήματα του Google Colab είναι ότι προσφέρει στους χρήστες δωρεάν πρόσβαση σε υπολογιστικούς πόρους όπως GPU και TPU. Αυτό σημαίνει ότι δίνεται η δυνατότητα σε οποιονδήποτε χρήστη να εκτελέσει heavy computing διεργασίες όπως την εκπαίδευση ενός μοντέλου μηχανικής μάθησης ανεξάρτητα από την ισχύ του δικού του μηχανήματος, αξιοποιώντας μόνο ένα πρόγραμμα περιήγησης.



Εικόνα 3.9: Runtimes Google Colab

Όπως φαίνεται και στην Εικόνα 3.9 υπάρχει η δυνατότητα επιλογής ανάμεσα σε διαφορετικά μοντέλα GPU και TPU αλλά και runtime (Python, R, Julia). Υπάρχουν ωστόσο περιορισμοί στην χρήση των πόρων, με τον κάθε χρήστη να έχει περιορισμένους πόρους GPU/TPU ανά ημέρα. Το πόσοι θα είναι αυτοί δεν είναι ξεκάθαρο καθώς εξαρτάται από τον φόρτο του συστήματος. Επίσης οι ανενεργές

συνεδρίες διακόπτονται ενώ ο μέγιστος χρόνος ενεργής συνεδρίας είναι οι 12 ώρες, αφού παρέλθουν αυτές διακόπτονται και ενεργές συνεδρίες.

Στο πλαίσιο της παρούσας εργασίας, το Google Colab λειτουργεί ως ιδανική πλατφόρμα για την εκτέλεση του κώδικα καθώς εξασφαλίζει ένα σταθερό περιβάλλον εργασίας, το οποίο όμως μπορεί να μεταβληθεί σε κάθε σημειωματάριο ανάλογα με τις ανάγκες. Επίσης μπορεί να ενσωματώσει γραφικά και οπτικοποιήσεις οι οποίες είναι ιδιαίτερα χρήσιμες στην εξαγωγή αναλυτικών δεδομένων. Ακόμα λόγω του διαδραστικού του χαρακτήρα ενδείκνυται για σκοπούς παρουσίασης ενώ προσφέρει εύκολο διαμοιρασμό. Τέλος η δωρεάν πρόσβαση σε GPU εξυπηρετεί ανάγκες που μπορεί να προκύψουν από μια βαριά διεργασία όπως NLP.

3.4.3 Δημιουργία προσαρμοσμένου loading script

Στο πλαίσιο της παρούσας εργασίας για την φόρτωση των δεδομένων δημιουργήθηκε ένα Python script το οποίο είναι προσαρμοσμένο στην δομή του dataset καθώς και στα πεδία του με σκοπό να προσφέρει αποδοτικότητα και **έλεγχο** στη φόρτωση μόνο των απαραίτητων δεδομένων από το αποθετήριο του Hugging Face. Για την ανάπτυξη του script χρησιμοποιήθηκε το περιβάλλον του Google Colab.

Το πρώτο βήμα είναι η εγκατάσταση των απαραίτητων βιβλιοθηκών για την υλοποίηση του script στο περιβάλλον του Google Colab. Ο τρόπος που επιλέχθηκε είναι με την χρήση του package manager pip [22]. Το pip είναι το επίσημο εργαλείο διαχείρισης πακέτων για την Python. Χρησιμοποιείται για την εγκατάσταση, ενημέρωση και αφαίρεση βιβλιοθηκών από το Python Package Index (PyPI) ή άλλες πηγές. Με την εντολή pip install package_name, μπορεί εύκολα να εγκατασταθεί το ζητούμενο module στο εκάστοτε περιβάλλον. Θα πρέπει να σημειωθεί ότι η έκδοση της Python που χρησιμοποιεί το Google Colab την χρονική στιγμή που διεκπεραιώνεται η παρούσα εργασία είναι η **3.11.11**. Οι βιβλιοθήκες που εγκαταστάθηκαν φαίνονται στην Εικόνα 3.10

```
[6]  # Install Required Packages

     !pip install datasets pandas
```

Εικόνα 3.10: Εγκατάσταση Python modules για το custom loading script

Η πιο σημαντική από αυτές είναι η datasets, η οποία αποτελεί την επίσημη βιβλιοθήκη για την φόρτωση dataset από το Hugging Face και περιέχει την συνάρτηση load_dataset που περιεγράφηκε στο υποκεφάλαιο 3.4.1. Η άλλη είναι η βιβλιοθήκη pandas [15] η οποία προσφέρει εύκολη, γρήγορη και ευέλικτη διαχείριση δεδομένων. Το επόμενο βήμα είναι η φόρτωση των απαραίτητων βιβλιοθηκών ή συγκεκριμένων συναρτήσεων, όπως φαίνεται στην Εικόνα 3.11

```
[2]  # Import required modules

     from datasets import load_dataset
     import pandas as pd
     from datetime import datetime
```

Εικόνα 3.11: Φόρτωση βιβλιοθηκών και συναρτήσεων για το custom loading script

Στη συνέχεια είναι το κομμάτι του script που περιέχει την λειτουργικότητα της φόρτωσης των δεδομένων. Πρόκειται για μια μέθοδο η οποία δέχεται τις παραμέτρους start_date και end_date οι οποίες είναι η αρχική ημερομηνία και η τελική ημερομηνία αντίστοιχα. Οι δύο αυτές ημερομηνίες ορίζουν

ουσιαστικά το χρονικό εύρος στο οποίο θα βρίσκονται τα δεδομένα τα οποία θα φορτωθούν. Το χρονικό φιλτράρισμα έχει γίνει με βάση το πεδίο date, το οποίο ουσιαστικά αποτελεί και το πεδίο με βάση το οποίο έγινε η κατανομή των CSV αρχείων σε φακέλους στο repository.

```python
# Function that loads the data by filtering the date given from user input

def load_csvs_from_huggingface(start_date, end_date):
    """
    Load only the necessary CSV files from a Hugging Face dataset repository.

    :param start_date: str, the start date in 'YYYY-MM-DD' format (inclusive)
    :param end_date: str, the end date in 'YYYY-MM-DD' format (inclusive)

    :return: pd.DataFrame, combined data from selected CSVs
    """
    huggingface_dataset_name = "amylonidis/PatClass2011"

    column_types = {
      "ucid": "string",
      "country": "category",
      "doc_number": "int64",
      "kind": "category",
      "lang": "category",
      "date": "int32",
      "application_date": "int32",
      "date_produced": "int32",
      "status": "category",
      "main_code": "string",
      "further_codes": "string",
      "ipcr_codes": "string",
      "ecla_codes": "string",
      "title": "string",
      "abstract": "string",
      "description": "string",
      "claims": "string",
      "applicants": "string",
      "inventors": "string",
  }
```

Εικόνα 3.12: Ορισμός column types στο custom loading script

Όπως φαίνεται και στην Εικόνα 3.12, το πρώτο κομμάτι του script είναι ο ορισμός των τύπων για όλα τα πεδία του dataset. Αυτό το βήμα είναι πολύ σημαντικό καθώς μειώνει σημαντικά την κατανάλωση της μνήμης RAM, επιταχύνει υπολογισμούς (ιδιαίτερα ομαδοποιήσεις, φιλτραρίσματα, ταξινομήσεις) και βελτιώνει σημαντικά την σταθερότητα του κώδικα. Πιο συγκεκριμένα ο τύπος numpy int32 χρησιμοποιεί 4 bytes άρα για 1 εκατομμύριο εγγραφές καταναλώνει περίπου 3.8 MB ενώ ο τύπος numpy int64 χρησιμοποιεί 8 bytes άρα για 1 εκατομμύριο εγγραφές καταναλώνει περίπου 7.6 MB. Για την περίπτωση της στήλης date που περιέχει τιμές τύπου YYYYMMDD (π.χ. 19870925) και οι οποίες δεν ξεπερνούν τα 2.5 δισεκατομμύρια, άρα χωρούν εύκολα σε int32 (εύρος: −2,147,483,648 έως +2,147,483,647) η χρήση αυτού του τύπου μειώνει την κατανάλωση μνήμης στο μισό χωρίς καμία απώλεια ακρίβειας. Το ίδιο ισχύει και για το application_date. Ένας ακόμα πολύ χρήσιμος τύπος που χρησιμοποιήθηκε είναι ο numpy category, ο οποίος χρησιμοποιείται για στήλες που περιέχουν επαναλαμβανόμενες διακριτές τιμές. Ο τύπος category λειτουργεί με βάση δύο βασικές δομές, τον lookup table που περιέχει μία φορά κάθε μοναδική τιμή (π.χ., "US", "FR", "DE") και τα διανύσματα δεικτών (codes) όπου για κάθε εγγραφή στον lookup table, αποθηκεύεται ένας ακέραιος δείκτης (π.χ., 0 για "US", 1 για "FR" κ.λπ.). Αυτοί οι δείκτες είναι int8, int16, int32 ανάλογα με το πλήθος των κατηγοριών. Επομένως ο τύπος χρησιμοποιεί 1 x N bytes, που είναι το συνολικό μέγεθος του lookup table, όπου κάθε τιμή είναι ένα κανονικό Python string (τύπου object). Για τα διανύσματα δείκτη χρησιμοποιεί στην χειρότερη περίπτωση (περισσότερες από 65.536 διακριτές κατηγορίες) 4 bytes. Άρα για 1 εκατομμύριο εγγραφές καταναλώνει 4 MB. Αν για ένα τέτοιο πεδίο όπως είναι το country (με

διακριτές τιμές "US", "FR", "DE") είχε χρησιμοποιηθεί ο default τύπος object, ο οποίος κατά μέσο όρο χρησιμοποιεί 50-60 bytes για 1 εκατομμύριο εγγραφές θα απαιτούνταν 50-60 MB μνήμης RAM. Άρα χρησιμοποιώντας τον τύπο category εξοικονομούνται έως και 90% της μνήμης για στήλες με λίγες επαναλαμβανόμενες τιμές. Εκτός από την κατανάλωση της μνήμης, η χρήση αυτού του τύπου έχει σημαντικά πλεονεκτήματα κατά την εκτέλεση κάποιων λειτουργιών του pandas. Παραδείγματα αποτελούν τα παρακάτω:

- Ομαδοποίηση (groupby): Είναι πολύ ταχύτερη, γιατί οι κατηγορίες είναι προ-καθορισμένες και αποθηκεύονται ως ακέραιοι δείκτες και το pandas δεν χρειάζεται να αναλύσει κείμενα (string) για κάθε τιμή
- Φιλτράρισμα: Οι συγκρίσεις με ==, != είναι πιο γρήγορες γιατί γίνονται σε ακέραιους δείκτες, όχι σε strings
- Ταξινόμηση (sort_values): Το sort_values() λειτουργεί πολύ πιο αποδοτικά γιατί το pandas ταξινομεί βάσει αριθμών, όχι αλφαβητικά ανά χαρακτήρα
- Μετρήσεις/Μοναδικότητα: Η μέτρηση συχνοτήτων (value_counts) ή μοναδικών τιμών (nunique) είναι πιο αποδοτική, καθώς βασίζεται σε περιορισμένο πλήθος κατηγοριών και όχι σε ελεύθερο κείμενο.

Γενικότερα, ο σαφής καθορισμός των τύπων έχει σαν αποτέλεσμα έναν πιο σταθερό κώδικα, αφού τα type errors που μπορούν να προκύψουν από την ιδιομορφία των δεδομένων, ειδικά σε μεγάλα dataset και με πολύ ελεύθερο κείμενο, είναι πολύ λιγότερα.

```
dataset_years = ['1978', '1979', '1980', '1981', '1982', '1983', '1984', '1985', '1986',
                 '1987', '1988', '1989', '1990', '1991', '1992', '1993', '1994', '1995',
                 '1996','1997', '1998', '1999', '2000', '2001', '2002','2003', '2004', '2005']

start_date_int = int(datetime.strptime(start_date, "%Y-%m-%d").strftime("%Y%m%d"))
end_date_int = int(datetime.strptime(end_date, "%Y-%m-%d").strftime("%Y%m%d"))

start_year, end_year = str(start_date_int)[:4], str(end_date_int)[:4]
given_years = [str(year) for year in range(int(start_year), int(end_year) + 1)]
matching_years = [f  for f in dataset_years for year in given_years if f==year]

if not matching_years:
    raise ValueError(f"No matching CSV files found in dataset for the given dates")
```

Εικόνα 3.13: Μεταβλητές custom loading script

Στην Εικόνα 3.13 φαίνεται η απαραίτητη προεργασία που γίνεται πριν το κομμάτι της φόρτωσης των δεδομένων. Αρχικά αποθηκεύονται στην μεταβλητή dataset_years όλα τα έτη που υπάρχουν σαν φάκελοι στο αποθετήριο του Hugging Face. Στη συνέχεια μετατρέπονται τα start_date και end_date που δίνονται από την χρήστη σε μορφή τύπου "1985-03-10" σε ακέραιο αριθμό της μορφής 19850310 και αποθηκεύονται στις μεταβλητές start_date_int και end_date_int. Αυτή η επεξεργασία γίνεται έτσι ώστε ο χρήστης να μπορεί να δώσει τις ημερομηνίες που θέλει σε εύκολα αναγνώσιμη μορφή. Έπειτα εξάγονται τα πρώτα 4 ψηφία από τα start_date_int και end_date_int που αποτελούν τα έτη, π.χ. αν start_date_int = 19850310, τότε start_year = '1985'. Τα start_year και end_year στην συνέχεια χρησιμοποιούνται για να δημιουργηθεί η λίστα given_years που περιέχει όλα τα έτη στο εύρος των ετών που έδωσε ο χρήστης. Για να δημιουργηθεί η λίστα matching_years που θα χρησιμοποιηθεί στην φόρτωση των δεδομένων, διατρέχεται η λίστα dataset_yeras και ελέγχεται ποια έτη ανήκουν ταυτόχρονα και στην λίστα given_years διατρέχοντας και αυτήν. Αν η λίστα matching_years είναι άδεια επιστρέφεται ένα μήνυμα λάθους στον χρήστη.

```
df_list = []
for year in matching_years:
    filepath = f"data/years/{year}/clefip2011_en_classification_{year}_validated.csv"

    try:
        dataset = load_dataset(huggingface_dataset_name, data_files=filepath, split="train")
        df = dataset.to_pandas().astype(column_types)
        mask = (df["date"] >= start_date_int) & (df["date"] <= end_date_int)
        df_filtered = df[mask].copy()


        if not df_filtered.empty:
            df_list.append(df_filtered)

        del df, dataset, df_filtered, mask
        gc.collect()

    except Exception as e:
        print(f"Error processing {filepath}: {e}")

return pd.concat(df_list, ignore_index=True) if df_list else pd.DataFrame()
```

Εικόνα 3.14: Φόρτωση δεδομένων στο custom loading script

Στην Εικόνα 3.14 φαίνεται το κομμάτι της φόρτωσης των δεδομένων από το Hugging Face. Διατρέχεται η λίστα με τα έτη που ζητήθηκαν από τον χρήστη και φτιάχνεται για κάθε έτος η αντίστοιχη διαδρομή αρχείου CSV μέσα στο αποθετήριο του Hugging Face. Έπειτα χρησιμοποιώντας την συνάρτηση load_dataset φορτώνονται τα δεδομένα από το CSV για το αντίστοιχο έτος χρησιμοποιώντας την μεταβλητή column_types που περιέχει τους τύπους για όλα τα πεδία, μετατρέπονται σε pandas αντικείμενο που λέγεται dataframe και αποθηκεύονται στη μεταβλητή df. Στη συνέχεια φιλτράρονται από το dataframe και επιλέγονται μόνο οι εγγραφές οι οποίες είναι στο χρονικό εύρος που έχει δώσει ο χρήστης. Το αποτέλεσμα αποθηκεύεται στην μεταβλητή df_filtered. Εφόσον η μεταβλητή df_filtered δεν είναι άδεια, προστίθεται στην λίστα df_list. Τέλος, σβήνονται οι αναφορές στις μεταβλητές df, dataset, df_filtered και mask, οι οποίες περιέχουν πολλά δεδομένα και δεν χρειάζονται πλέον, και καλείται χειροκίνητα η μέθοδος του garbage collection ώστε να απελευθερωθεί δεσμευμένη μνήμη. Αν προκύψει κάποιο σφάλμα κατά την φόρτωση επιστρέφεται μήνυμα λάθους στον χρήστη. Αφού ολοκληρωθεί αυτή η διαδικασία για όλα τα έτη, τότε δημιουργείται ένα συγκεντρωτικό pandas dataframe με όλα τα δεδομένα από την λίστα df_list το οποίο και επιστρέφεται.

Η φόρτωση των δεδομένων γίνεται καλώντας την συνάρτηση load_csvs_from_huggingface και περνώντας σαν παραμέτρους την ημερομηνία έναρξης και την ημερομηνία λήξης του χρονικού εύρους όπως φαίνεται στην Εικόνα 3.15. Για το συγκεκριμένο εύρος που φαίνεται στην Εικόνα 3.15 δηλαδή από 1984-07-09 μέχρι και 1992-05-10 και πιο συγκεκριμένα περίπου 5 GB τα οποία αντιστοιχούν σε 58887 εγγραφές, απαιτούνται περίπου 4 λεπτά για την φόρτωση και 6.5 GB RAM.

```
start_date = "1984-07-09"
end_date = "1992-05-10"

df = load_csvs_from_huggingface(start_date, end_date)
```

```
Εμφάνιση μη ορατής εξόδου
```

```
[ ] print(df.shape)
```

```
(58887, 19)
```

Εικόνα 3.15: Χρήση του custom loading script

Η παραπάνω μεθοδολογία επιτρέπει αποδοτική και στοχευμένη φόρτωση δεδομένων από ένα μεγάλο χρονικά κατανεμημένο αποθετήριο. Ορίζοντας συγκεκριμένους τύπους για τα δεδομένα, μειώνεται σημαντικά η κατανάλωση μνήμης, προσφέρεται ενοποιημένη αναπαράσταση και ορθότερα αποτελέσματα σε αναλύσεις που θα ακολουθήσουν. Ακόμα, το script περιλαμβάνει χειρισμό εξαιρέσεων (exception handling), προσφέροντας ασφάλεια στη διαδικασία φόρτωσης. Σε περίπτωση αποτυχίας ανάκτησης ενός αρχείου, το σύστημα συνεχίζει να λειτουργεί με τα υπόλοιπα διαθέσιμα αρχεία, διατηρώντας τη λειτουργικότητα. Επίσης, η χρήση φιλτραρίσματος κατά το αρχικό στάδιο μειώνει τον όγκο δεδομένων που απαιτείται να αναλυθεί, εξοικονομώντας υπολογιστικό χρόνο και μνήμη. Τέλος, η συνάρτηση μπορεί εύκολα να ενσωματωθεί σε άλλα pipelines δεδομένων, διατηρώντας πλήρως παραμετροποιήσιμο και επαναχρησιμοποιήσιμο χαρακτήρα. Η δομή του script είναι σαφής και επεκτάσιμη. Για παράδειγμα, μελλοντική προσθήκη νέων ετών ή επιπλέον φίλτρων (όπως χώρα ή κωδικός ταξινόμησης) μπορούν να υλοποιηθούν με μικρές μόνο αλλαγές.

Κεφάλαιο 4ο: Ανάλυση συλλογής CLEF-IP2011

Η ανάλυση δεδομένων πάνω σε σύνολα πατεντών αποτελεί βασικό εργαλείο για την κατανόηση των τεχνολογικών τάσεων, την αξιολόγηση της καινοτομίας σε εθνικό και διεθνές επίπεδο και την υποστήριξη στρατηγικών αποφάσεων σε βιομηχανίες υψηλής τεχνολογίας. Μέσα από την εξαγωγή analytics, μπορούν να αποκαλυφθούν πρότυπα δραστηριότητας, γεωγραφικής κατανομής, εξέλιξης τεχνολογικών τομέων, αλλά και δυναμική οργανισμών ή επιμέρους εφευρετών.

Στο κεφάλαιο αυτό θα αναλυθεί η εξαγωγή όλων των αναλυτικών δεδομένων (analytics) που έγινε στο dataset και οι οπτικοποιήσεις αυτών, το κομμάτι του γράφου δηλαδή που φαίνεται στην Εικόνα 4.1

Εικόνα 4.1: Google Colab Analytics στον Γράφο

4.1 Εξαγωγή αναλυτικών δεδομένων και οπτικοποιήσεις

Η διαδικασία της επεξεργασίας των δεδομένων και της εξαγωγής των analytics έγινε στο περιβάλλον του Google Colab σε διαφορετικά notebooks για το κάθε πεδίο προς ανάλυση. Για τις αναλύσεις επιλέχθηκε ένα υποσύνολο του dataset, το οποίο είναι τα δεδομένα από 1987-07-09 μέχρι και 1993-12-12 που αντιστοιχούν σε 70354 εγγραφές και περίπου 5.5 GB. Για την φόρτωση των δεδομένων χρησιμοποιήθηκε το custom loading script.

4.1.1 Ανάλυση Dates

Το χρονικό πεδίο που αναλύθηκε είναι το date, που αντιστοιχεί στην ημερομηνία δημοσίευσης της κάθε πατέντας. Φορτώνοντας τα δεδομένα με το custom loading script στην μεταβλητή df τύπου pandas dataframe πολύ εύκολα μπορεί να υπολογισθεί ο μέσος όρος, ο διάμεσος και η τυπική απόκλιση σε επίπεδο έτους και δεκαετίας όπως φαίνεται στην Εικόνα 4.2

```
[9]  mean_year = df['year'].mean()
     median_year = df['year'].median()
     std_year = df['year'].std()

     print(f" Year - Mean: {mean_year}, Median: {median_year}, Standard Deviation: {std_year}")


     mean_decade = df['decade'].mean()
     median_decade = df['decade'].median()
     std_decade = df['decade'].std()

     print(f"Decade - Mean: {mean_decade}, Median: {median_decade}, Standard Deviation: {std_decade}")

 Year - Mean: 1991.501023395969, Median: 1992.0, Standard Deviation: 1.5436472643343058
Decade - Mean: 1988.7898342667083, Median: 1990.0, Standard Deviation: 3.2614885294568743
```

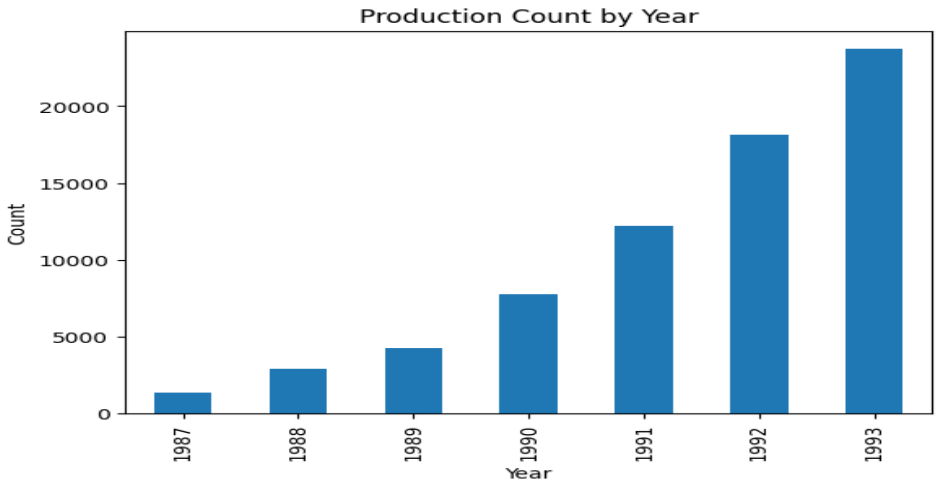
Εικόνα 4.2: Μέσος όρος, ενδιάμεσος και τυπική απόκλιση date

Οι μετρικές αυτές παρότι χρήσιμες δεν είναι πολύ ενδεικτικές για την κατανομή των δεδομένων ή για την εμφάνιση κάποιας τάσης. Για αυτές τις περιπτώσεις ιδιαίτερα χρήσιμα είναι τα γραφήματα, τα οποία μπορούν να δημιουργηθούν με την χρήση της βιβλιοθήκης pyplot και πιο συγκεκριμένα με την συνάρτηση matplotlib [23]. Για την δημιουργία του γραφήματος που απεικονίζει το πλήθος των πατεντών ανά έτος δημοσίευσης υλοποιήθηκε το κομμάτι κώδικα που φαίνεται στην Εικόνα 4.3

```
label = f"Production Count by {examined_sub_analytic_label}"

# Plot data counts over years
df[examined_sub_analytic].value_counts().sort_index().plot(kind='bar')
plt.xlabel(examined_sub_analytic_label)
plt.ylabel('Count')
plt.title(label)
plt.show()
```

Εικόνα 4.3: Κώδικας γραφήματος πλήθους πατεντών ανά έτος δημοσίευσης

Πρώτα φτιάχνεται ο τίτλος του γραφήματος δυναμικά χρησιμοποιώντας την μεταβλητή examined_sub_analytic_label. Έπειτα χρησιμοποιώντας την συνάρτηση matplotlib απεικονίζονται σε ένα γράφημα μπάρας, το ταξινομημένο άθροισμα πατεντών με βάση την μεταβλητή examined_sub_analytic , η οποία στην συγκεκριμένη περίπτωση έχει οριστεί να είναι το year. Έπειτα ορίζονται οι τιμές που θα υπάρχουν στους άξονες x και y και ο τίτλος του γραφήματος. Τέλος εμφανίζεται το γράφημα, με την εντολή plt.show(), όπως φαίνεται και στην Εικόνα 4.4



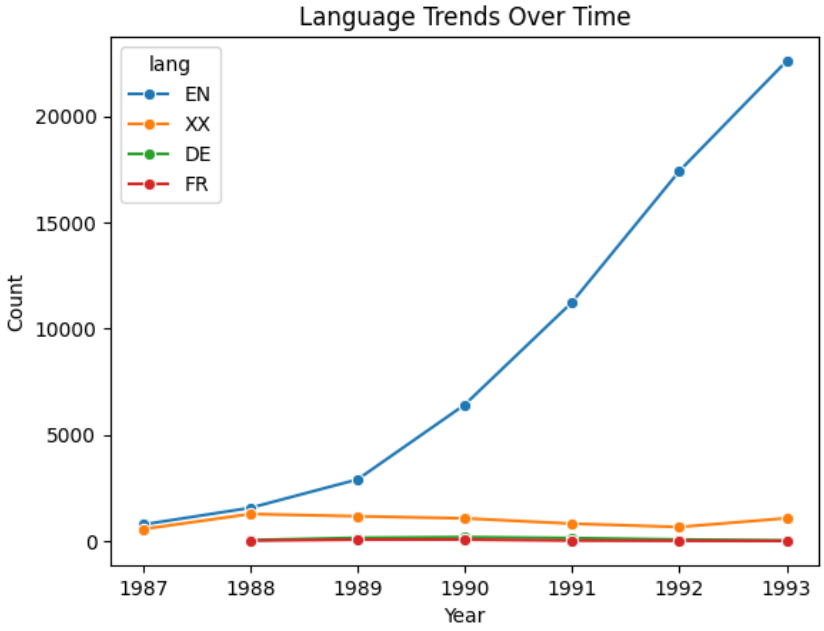
Εικόνα 4.4: Γράφημα πλήθους πατεντών ανά έτος δημοσίευσης

Από αυτό το γράφημα φαίνεται η σταθερή και σταδιακή αύξηση του πλήθους μέσα στον χρόνο, με την τριετία 1987-1990 να παρουσιάζει κάθε χρόνο διπλάσιο αριθμό πατεντών από τον προηγούμενο. Το μεγαλύτερο άλμα είναι από το 1991 (12,300) στο 1992 (18,200), μια αύξηση 5.900 πατεντών (≈48% αύξηση). Αυτός ο ρυθμός ανάπτυξης καταδεικνύει περίοδο ταχείας επέκτασης της πατεντιακής δραστηριότητας, πιθανώς οδηγούμενη από την ταχεία τεχνολογική ανάπτυξη. Για την δημιουργία ενός time series γραφήματος που αναλύει το πλήθος των πατεντών ανά γλώσσα υλοποιήθηκε ο κώδικας που φαίνεται στην Εικόνα 4.5

```
label = f"{examined_analytic_to_compare_label} Trends Over Time"

# Group data by year and another field
grouped = df.groupby([examined_sub_analytic, examined_analytic_to_compare]).size().reset_index(name='count')

# Plot
sns.lineplot(data=grouped, x=examined_sub_analytic, y='count', hue=examined_analytic_to_compare, marker='o')
plt.title(label)
plt.xlabel(examined_sub_analytic_label)
plt.ylabel("Count")
plt.show()
```

Εικόνα 4.5: Κώδικας γραφήματος πλήθους πατεντών ανά γλώσσα, ανά έτος

Η διαφορά με το προηγούμενο γράφημα είναι ότι εδώ χρησιμοποιήθηκε και η βιβλιοθήκη seaborn [24], η οποία βασίζεται στην matplotlib και παρέχει μια υψηλού επιπέδου και φιλική προς τον χρήστη διεπαφή για δημιουργία γραφημάτων. Αρχικά, αποθηκεύεται στην μεταβλητή grouped το dataframe με στήλες το year και το lang (έχουν οριστεί στις μεταβλητές examined_sub_analytic και examined_analytic_to_compare αντίστοιχα) και αφού υπολογισθεί το άθροισμα (count) κάθε τέτοιου ζευγαριού αποθηκεύεται και αυτό σαν στήλη. Στη συνέχεια με την χρήση της βιβλιοθήκης seaborn δημιουργείται το γράφημα, απεικονίζεται το άθροισμα σε σχέση με τον χρόνο (year) ζωγραφίζοντας μια γραμμή για κάθε τιμή της μεταβλητής lang. Έπειτα ορίζονται οι τιμές που θα υπάρχουν στους άξονες x και y και ο τίτλος του γραφήματος και εμφανίζεται το γράφημα που φαίνεται στην Εικόνα 4.6.



Εικόνα 4.6: Γράφημα πλήθους πατεντών ανά γλώσσα, ανά έτος

Από το γράφημα στην Εικόνα 4.6 φαίνεται ξεκάθαρα η κυριαρχία της αγγλικής γλώσσας με ραγδαία αύξηση από τις 800 στις 17.400 πατέντες η οποία είναι σταθερή στο πέρασμα του χρόνου και καταδεικνύει μια ξεκάθαρη τάση. Από την άλλη οι πατέντες στην γαλλική, γερμανική και οποιαδήποτε άλλη γλώσσα έπιασαν το ζενίθ τους ανάμεσα σε 1988–1989 και στη συνέχεια κυμάνθηκαν σε χαμηλά σύνολά. Μια ερμηνεία αυτού είναι ότι οι αιτούντες επέλεξαν την αγγλική γλώσσα περισσότερο για ευρύτερη προστασία. Αξίζει να σημειωθεί ότι το γράφημα αυτό μπορεί να δημιουργηθεί για τα πεδία country, status ή οποιοδήποτε πεδίο με την μορφή κατηγορίας, ορίζοντας το στην μεταβλητή examined_analytic_to_compare και ξανατρέχοντας το συγκεκριμένο κομμάτι κώδικα.

4.1.2 Ανάλυση Applicant – Inventor Names

Τα πεδία που αναλύθηκαν είναι το appplicants και το inventors, τα ονόματα των αιτούντων και των εφευρετών αντίστοιχα. Θα επιδειχθούν μόνο το αποτελέσματα για το applicants, αλλά τρέχοντας τον ίδιο κώδικα αλλάζοντας την μεταβλητή examined_analytic μπορούν να εξαχθούν τα αποτελέσματα και για το inventors. Αφού φορτωθούν τα δεδομένα στην μεταβλητή df γίνεται η προ-επεξεργασία των δεδομένων όπως φαίνεται στην Εικόνα 4.7

```
df[examined_analytic] = df[examined_analytic].fillna("").astype(str)

df["field_after_cleanup"] = df[examined_analytic].apply(lambda x: x.split(";"))

all_names = [name.strip() for sublist in df["field_after_cleanup"] for name in sublist]
unique_names = set(all_names)
print(f"Total Unique {examined_analytic_label}: {len(unique_names)}")


name_counts = Counter(all_names)
```

```
Total Unique Applicants: 42444
```

Εικόνα 4.7: Κώδικας προ-επεξεργασίας για τα πεδία applicants και inventors

Αρχικά αντικαθίστανται οι απούσες εγγραφές με κενό string και μετατρέπονται όλες οι εγγραφές της εξεταζόμενης στήλης σε string. Έπειτα για κάθε εγγραφή, το string που περιέχει τα ονόματα διαχωριζόμενα από τον χαρακτήρα ; μετατρέπεται σε μια λίστα που περιέχει αυτά τα ονόματα. Στη συνέχεια διατρέχοντας όλες τις εγγραφές, αποθηκεύονται όλα τα ονόματα στην μεταβλητή all_names. Για την καταμέτρηση των ξεχωριστών ονομάτων χρησιμοποιείται η συνάρτηση set που αφαιρεί τα διπλότυπα και το αποτέλεσμα αποθηκεύεται στην μεταβλητή unique names τα οποία είναι 42.444. Ακόμα χρησιμοποιείται η συνάρτηση Counter στην μεταβλητή all_names ώστε να καταμετρηθεί το σύνολο των εγγραφών για το κάθε όνομα και το αποτέλεσμα αποθηκεύεται στην μεταβλητή name_counts. Στην Εικόνα 4.8 φαίνεται ο κώδικας για την εμφάνιση των 10 ονομάτων με τις περισσότερες εγγραφές.

```
name_counts_df = pd.DataFrame(name_counts.items(), columns=[f"{examined_analytic_label}", "Patent Count"])
name_counts_df = name_counts_df[name_counts_df[f"{examined_analytic_label}"] != ""]
top_names = name_counts_df.sort_values(by="Patent Count", ascending=False).head(10)
print(top_names)
```

	Applicants	Patent Count
21	INTERNATIONAL BUSINESS MACHINES CORPORATION	925
90	E.I. DU PONT DE NEMOURS AND COMPANY	791
67	N.V. PHILIPS' GLOEILAMPENFABRIEKEN	692
4355	EASTMAN KODAK COMPANY	571
117	MINNESOTA MINING AND MANUFACTURING COMPANY	565
4354	EASTMAN KODAK CO	553
2314	DU PONT	505
113	KABUSHIKI KAISHA TOSHIBA	498
68	SHELL INTERNATIONALE RESEARCH MAATSCHAPPIJ B.V.	464
9	THE DOW CHEMICAL COMPANY	413

Εικόνα 4.8: Κώδικάς για την εμφάνιση 10 ονομάτων με τις περισσότερες εγγραφές και αποτέλεσμα για applicants

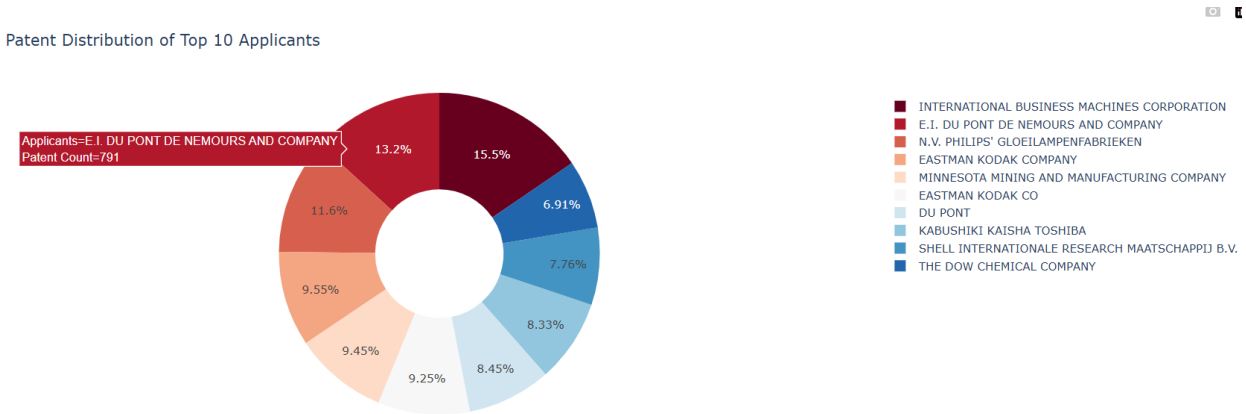
Στην μεταβλητή name_counts_df αποθηκεύεται ένα pandas dataframe το οποίο έχει δύο στήλες, το όνομα και τον αριθμό των εμφανίσεων του. Έπειτα αφού αφαιρεθούν οι κενές εγγραφές, το dataframe αυτό ταξινομείται σε φθίνουσα σειρά με βάση τον αριθμό των εμφανίσεων του κάθε ονόματος και επιλέγονται οι πρώτες 10 εγγραφές οι οποίες αποθηκεύονται στην μεταβλητή top_names. Το

αποτέλεσμα φαίνεται στην Εικόνα 4.8 με την IBM να είναι πρώτη και την E.I. DU PONT DE NEMOURS AND COMPANY δεύτερη, η οποία ωστόσο εμφανίζεται και ως DU PONT χαμηλότερα, χωρίς πάντως οι αιτούντες να έχουν τεράστιες διαφορές μεταξύ τους.

```
title = f"Patent Distribution of Top 10 {examined_analytic_label}"

fig = px.pie(
    top_names,
    values="Patent Count",
    names=f"{examined_analytic_label}",
    title=title,
    hole=0.4,
    color_discrete_sequence=px.colors.sequential.RdBu,
)

fig.show()
```

Εικόνα 4.9: Κώδικας απεικόνισης κατανομής ονομάτων σε piechart

Για την απεικόνιση της κατανομής των ονομάτων σε piechart χρησιμοποιήθηκε η βιβλιοθήκη plotly [25], η οποία επιτρέπει την δημιουργία διαδραστικών (μέσω browser) γραφημάτων. Για παράδειγμα ο χρήστης κλικάροντας μια συγκεκριμένη επιλογή από το legend του γραφήματος μπορεί να το αφαιρέσει από το γράφημα on the fly. Για το piechart χρησιμοποιήθηκε η συνάρτηση pie και περάστηκε σαν παράμετρος η μεταβλητή top_names που περιέχει τα δεδομένα ενώ οι υπόλοιπες παράμετροι που φαίνονται στην Εικόνα 4.9 σχετίζονται με τα γραφιστικά, δηλαδή τον τίτλο, τα χρώματα του piechart και το μέγεθος.



Εικόνα 4.10: Piechart κατανομής ονομάτων για applicants

Όπως φαίνεται και στην Εικόνα 4.10 κάνοντας hover πάνω από μια περιοχή του γραφήματος εμφανίζεται το όνομα και ο αριθμός των πατεντών που αντιστοιχούν σε αυτή την περιοχή. Για την ανάλυση των ονομάτων σε σχέση με την γλώσσα χρειάστηκε να γίνει περεταίρω επεξεργασία, όπως φαίνεται στην Εικόνα 4.11

```
[67] df['names_list'] = (df[examined_analytic].apply(lambda cell: [name.strip() for name in cell.split(';') if name.strip()]))


    df_exploded = (df.explode('names_list').rename(columns={'names_list': 'name'}))
    combined = (df_exploded.groupby(['name', examined_analytic_to_compare]).size().reset_index(name='Patent Count'))


    pivot = (combined.pivot(index='name', columns=examined_analytic_to_compare, values='Patent Count').fillna(0))

    pivot = (
        pivot
          .assign(Total=lambda d: d.sum(axis=1))
          .sort_values('Total', ascending=False)
          .drop(columns='Total')
    )
```

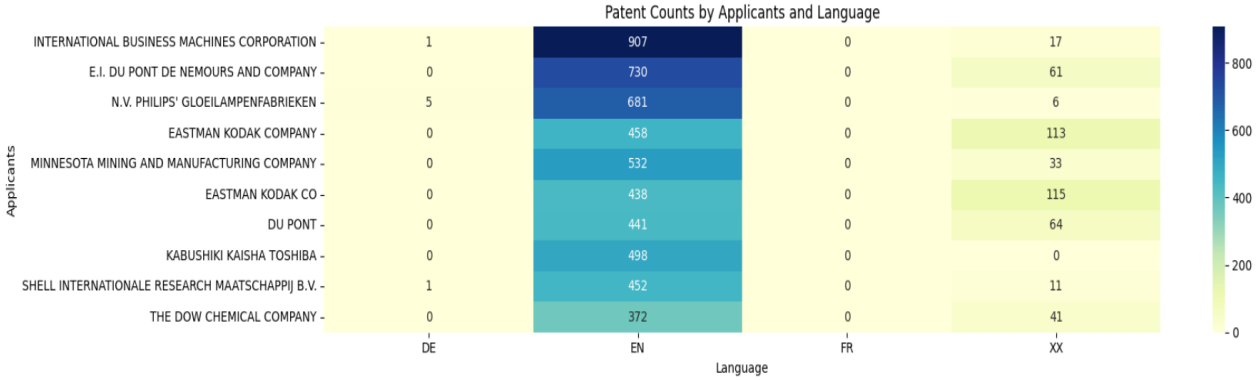
Εικόνα 4.11: Κώδικας ανάλυσης ονομάτων σε σχέση με την γλώσσα

Πρώτα, χρησιμοποιείται η συνάρτηση apply ώστε να διατρεχθεί κάθε τιμή της στήλης που έχει οριστεί στην μεταβλητή examined_analytic (εδώ το applicants) και για όλα τα στοιχεία που χωρίζονται με τον χαρακτήρα ; στην τιμή της στήλης να αφαιρεθούν οι επιπλέον κενοί χαρακτήρες και να μην συμπεριληφθούν κενά strings. Το αποτέλεσμα αυτής της επεξεργασίας αποθηκεύεται στο df[‘names_list’]. Έπειτα χρησιμοποιείται η συνάρτηση explode η οποία μετασχηματίζει κάθε λίστα του κάθε κελιού της στήλης names_list σε ξεχωριστές εγγραφές για κάθε στοιχείο της λίστας και το αποτέλεσμα αποθηκεύεται στην μεταβλητή df_exploded. Στη συνέχεια χρησιμοποιείται η συνάρτηση group_by ώστε να κατηγοριοποιηθεί το exploded με βάση το name και την μεταβλητή examined_analytic_to_compare (εδώ είναι το lang αφού η ανάγκη είναι να γίνει ανάλυση συγκριτικά με την γλώσσα) και γίνεται χρήση του size ώστε να καταμετρηθούν οι εμφανίσεις του κάθε ζευγαριού name και examined_analytic_to_compare. Με το reset_index επιστρέφεται ένα dataframe στην μεταβλητή combined, με τον αριθμό των εμφανίσεων να αποθηκεύεται στην στήλη Patent Count. Το επόμενο βήμα είναι να δημιουργηθεί ένας πίνακας με την χρήση της συνάρτησης pivot, που έχει σαν στήλες τις τιμές του examined_analytic_to_compare, σαν γραμμές τα ονόματα και σαν τιμές τον αριθμό των πατεντών. Τέλος τα περιεχόμενα αυτού του πίνακα, που βρίσκονται στην μεταβλητή pivot, ταξινομούνται με βάση τον συνολικό αριθμό των πατεντών κάθε γραμμής αξιοποιώντας την προσωρινή στήλη Total η οποία μετά την ταξινόμηση διαγράφεται. Η μεταβλητή pivot μετά από αυτή την επεξεργασία μπορεί να χρησιμοποιηθεί για την δημιουργία ενός γραφήματος που να περιλαμβάνει όλα αυτά τα στοιχεία όπως φαίνεται στην Εικόνα 4.12

```
title = f"Patent Counts by {examined_analytic_label} and {examined_analytic_to_compare_label}"

top_rows= pivot.head(10)

plt.figure(figsize=(20, 8))
sns.heatmap(top_rows, annot=True, fmt=".0f", cmap="YlGnBu", cbar=True)
plt.title(title)
plt.xlabel(f"{examined_analytic_to_compare_label}")
plt.ylabel(f"{examined_analytic_label}")
plt.tight_layout()
plt.show()
```

Εικόνα 4.12: Κώδικας γραφήματος ονομάτων σε σύγκριση με την γλώσσα

Από τον πίνακα pivot επιλέγονται οι πρώτες δέκα εγγραφές και αποθηκεύονται στην μεταβλητή top_rows η οποία και χρησιμοποιείται για να δημιουργηθεί το γράφημα τύπου heatmap με την συνάρτηση seaborn[24] το οποίο φαίνεται στην Εικόνα 4.13



Εικόνα 4.13: Heatmap applicants και language

Θα πρέπει να σημειωθεί ότι το pivot δεν είναι αποκλειστικά κατασκευασμένο για να χρησιμοποιηθεί στο συγκεκριμένο heatmap ή μόνο με την συνάρτηση seaborn αλλά μπορεί κάλλιστα να χρησιμοποιηθεί και από την βιβλιοθήκη matplotlib. Ακόμα, αντί για το lang μπορεί να αναλυθεί οποιοδήποτε πεδίο έχει την μορφή κατηγορίας, ορίζοντας το στην μεταβλητή examined_analytic_to_compare και ξανατρέχοντας το συγκεκριμένο κομμάτι κώδικα.

4.1.3 Ανάλυση Codes

Το πεδίο που αναλύθηκε είναι το main_code, ο κώδικας που αναπτύχθηκε ωστόσο υποστηρίζει όλα τα πεδία κωδικών του dataset, δηλαδή τα further_codes, ipcr_codes και ecla_codes. Θα πρέπει να σημειωθεί ότι το πεδίο main_code περιέχει πάντα έναν κωδικό, ενώ τα υπόλοιπα μια λίστα κωδικών όπου το κάθε στοιχείο-κωδικός χωρίζεται από τα υπόλοιπα με κόμμα. Τα βήματα που ακολουθούν της φόρτωσης των δεδομένων φαίνονται στην Εικόνα 4.14

```python
def split_codes(cell):
    # Guard against NaN
    if pd.isna(cell) or not cell.strip():
        return []
    # Split on comma, strip whitespace
    parts = [part.strip() for part in cell.split(',')]
    # Drop any empty
    return [part for part in parts if part]
```

```python
df['codes_list'] = df[examined_analytic].apply(split_codes)
level=None
if level:
  all_codes = [code[level-1] for sublist in df['codes_list'] for code in sublist]
else:
  all_codes = [code for sublist in df['codes_list'] for code in sublist]
code_counts = Counter(all_codes)

top_n = 10
code_counts = (
    pd.DataFrame(code_counts.items(), columns=[examined_analytic_label, 'Patent Count'])
      .sort_values('Patent Count', ascending=False)
      .head(top_n)
)
```

Εικόνα 4.14: Κώδικας επεξεργασίας για στήλη τύπου codes

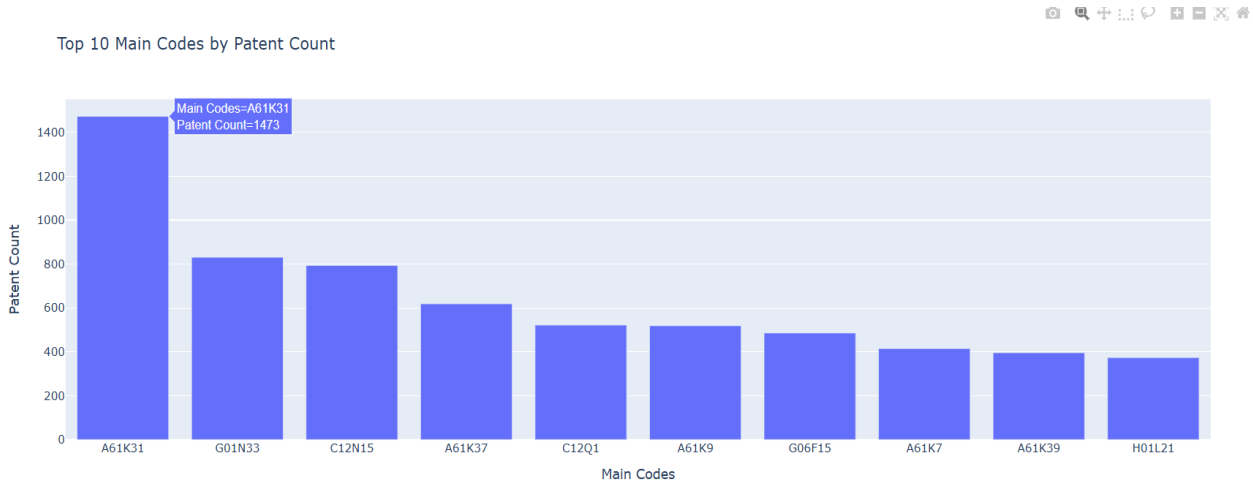
Αρχικά δημιουργήθηκε η μέθοδος split_codes, η οποία παίρνει σαν παράμετρο μια τιμή της στήλης τύπου codes και σπάει αυτή την τιμή η οποία είναι ένα ενιαίο string σε μια λίστα. Σαν αποτέλεσμα επιστρέφει μόνο τα στοιχεία της κάθε λίστας που περιέχουν τιμές, απορρίπτοντας τα κενά στοιχεία. Η

μέθοδος αυτή καλείται σε όλες τις εγγραφές της εξεταζόμενης στήλης χρησιμοποιώντας την συνάρτηση apply. Έπειτα όλα τα στοιχεία για όλες τις εγγραφές της εξεταζόμενης στήλης του dataset ενώνονται σε μια ενιαία λίστα και αποθηκεύονται στην μεταβλητή all_codes, κρατώντας μόνο τον αριθμό των γραμμάτων που επιλέγονται ανάλογα με το δηλωθέν επίπεδο στην μεταβλητή level. Η εκτέλεση που φαίνεται είναι χωρίς να επιλεχθεί επίπεδο. Με την συνάρτηση Counter μετρούνται όλες οι μοναδικές τιμές της all_codes και αποθηκεύονται στην μεταβλητή code_counts. Τέλος για να χρησιμοποιηθούν σε γραφήματα τα δεδομένα code_counts μετατρέπονται σε ένα pandas dataframe, με στήλες τον κωδικό και τον αντίστοιχο αριθμό εμφανίσεων αυτού του κωδικού, ταξινομώντας τις τιμές με φθίνουσα σειρά με βάση τον αριθμό εμφανίσεων και επιλέγονται οι 10 κορυφαίοι κωδικοί. Ο κώδικας για την γραφική αναπαράσταση φαίνεται στην Εικόνα 4.15

```
title = f"Top 10 {examined_analytic_label} by Patent Count"

fig = px.bar(
    code_counts,
    x=examined_analytic_label,
    y="Patent Count",
    title=title,
    labels={examined_analytic_label: examined_analytic_label},
)

fig.show()
```

Εικόνα 4.15: Κώδικας γραφήματος bar για στήλη τύπου codes

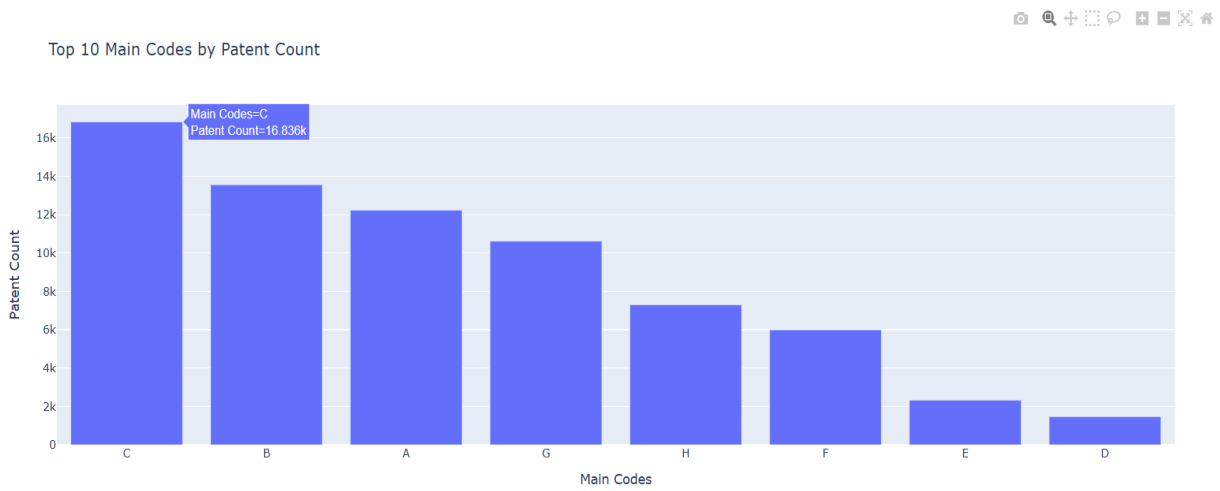
Όπως φαίνεται στην Εικόνα 4.15 χρησιμοποιείται η βιβλιοθήκη plotly[25], καλώντας την μέθοδο bar με παράμετρο την μεταβλητή code_counts που περιέχει το dataframe με τους 10 κορυφαίους σε εμφανίσεις κωδικούς. Όπως και στα προηγούμενα γραφήματα ο τίτλος, οι ετικέτες και οι τιμές του άξονα x ή y ορίζονται δυναμικά με βάση την μεταβλητή examined_analytic_label.



Εικόνα 4.16: Γράφημα bar για την στήλη τύπου codes

Το γράφημα αυτό δείχνει μια ξεκάθαρη διαφορά του πρώτου κωδικού από τους υπόλοιπους. Αναλύοντας ένα προς ένα τους κωδικούς γίνεται αντιληπτό ότι το 80% αυτών αναφέρονται στον τομέα της υγείας (βιοτεχνολογία, φαρμακευτική). Παρότι μπορούν να εξαχθούν χρήσιμα συμπεράσματα από αυτή την ανάλυση, έχει ενδιαφέρον να ερευνηθεί αν τα αποτελέσματα θα είναι παρόμοια επιλέγοντας

επίπεδο. Ορίζοντας λοιπόν την μεταβλητή level = 1, διαλέγοντας δηλαδή το πρώτο επίπεδο και ξανατρέχοντας τον κώδικα το γράφημα που παράγεται φαίνεται στην Εικόνα 4.17



Εικόνα 4.17: Γράφημα κωδικών για το πρώτο επίπεδο

Από το γράφημα στην Εικόνα 4.17 που δείχνει την πρώτη κατηγορία κωδικών να είναι το C, φαίνεται πως σε γενικότερο πλαίσιο στο dataset έχει ισχυρή παρουσία ο τομέας της Χημείας ενώ ακολουθούν οι κατηγορίες Performing Operations, Ανθρώπινες Ανάγκες και η Φυσική. Φαίνεται λοιπόν πως εκτελώντας τον κώδικα για διαφορετικά επίπεδα γίνονται γνωστές διαφορετικές πτυχές του dataset.

4.1.4 Ανάλυση Section Title

Το πεδίο που αναλύθηκε είναι το title το οποίο αναφέρεται στον τίτλο της κάθε πατέντας. Μετά την φόρτωση των δεδομένων υπολογίσθηκε ο μέσος όρος και ο διάμεσος των λέξεων του τίτλου όπως φαίνεται στην Εικόνα 4.18

```
df["word_count"] = df[examined_analytic].apply(lambda x: len(str(x).split()))

mean_word_count = df["word_count"].mean()
median_word_count = df["word_count"].median()

print(f"Mean number of words in {examined_analytic_label}: {mean_word_count}")
print(f"Median number of words in {examined_analytic_label}: {median_word_count}")
```

```
Mean number of words in title: 6.894519146032919
Median number of words in title: 6.0
```

Εικόνα 4.18: Μέσος όρος και διάμεσος λέξεων του τίτλου

Με την συνάρτηση apply διατρέχονται όλες οι εγγραφές του dataset και για κάθε εγγραφή δημιουργείται μια λίστα όπου κάθε λέξη είναι ένα στοιχείο της λίστας. Ο διαχωρισμός των λέξεων γίνεται με βάση το whitespace. Με τις συναρτήσεις mean και median υπολογίζονται εύκολα ο μέσος όρος και ο διάμεσος των λέξεων όπως φαίνονται στην Εικόνα 4.18.

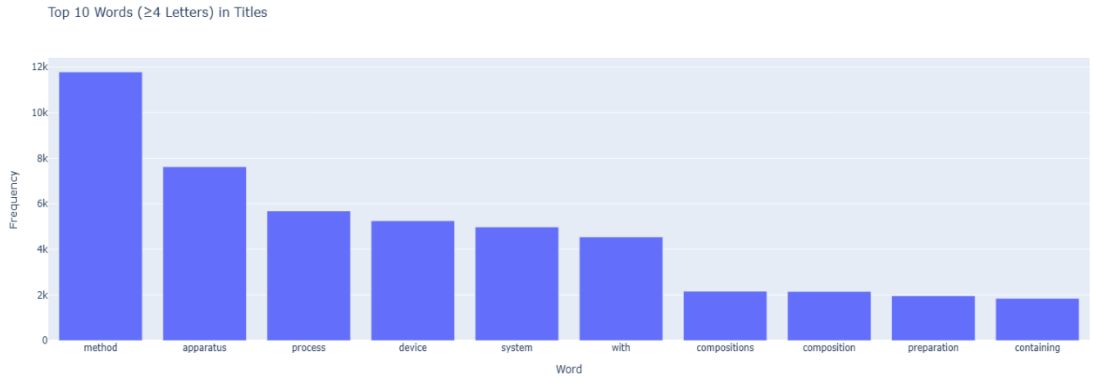
```
[ ]  def extract_long_words(text):
         return re.findall(r"[A-Za-z]{4,}", str(text).lower())
```

```
     all_words = (
         df[examined_analytic]
           .dropna()
           .apply(extract_long_words)
           .explode()
     )

     word_counts = Counter(all_words)
```

```
[ ]  n = 10
     top_n = word_counts.most_common(n)
     top_df = pd.DataFrame(top_n, columns=['word','count'])


     title = f"Top {n} Words (≥4 Letters) in {examined_analytic_label}"
     x_label = "word"
     y_label = "count"

     fig = px.bar(
         top_df,
         x=x_label,
         y=y_label,
         title="Top 10 Words (≥4 Letters) in Titles",
         labels={x_label:'Word',y_label:'Frequency'}
     )

     fig.show()
```

Εικόνα 4.19: Κώδικας απεικόνισης των 10 δημοφιλέστερων λέξεων με 4 γράμματα και πάνω

Ο κώδικάς της Εικόνας 4.19 παράγει ένα γράφημα με τις 10 δημοφιλέστερες λέξεις με μήκος άνω των τεσσάρων γραμμάτων. Το μήκος αυτό επιλέχθηκε με σκοπό να αποφευχθούν μικρές λέξεις (π.χ. the, on, at) οι οποίες δεν προσφέρουν αξία στην ανάλυση. Δημιουργήθηκε η μέθοδος extract_long_words η οποία με την χρήση regular expressions εντοπίζει string με μήκος μεγαλύτερο από 4 χαρακτήρες. Η συνάρτηση εφαρμόζεται σε όλες τις εγγραφές και αφού αφαιρεθούν οι κενές τιμές μετριέται το πλήθος κάθε λέξης. Στη συνέχεια επιλέγονται οι 10 με τα μεγαλύτερα πλήθη και δημιουργείται ένα dataframe με δύο στήλες, την λέξη και το αντίστοιχο πλήθος, το οποίο αναπαρίσταται γραφικά με την βιβλιοθήκη plotly.



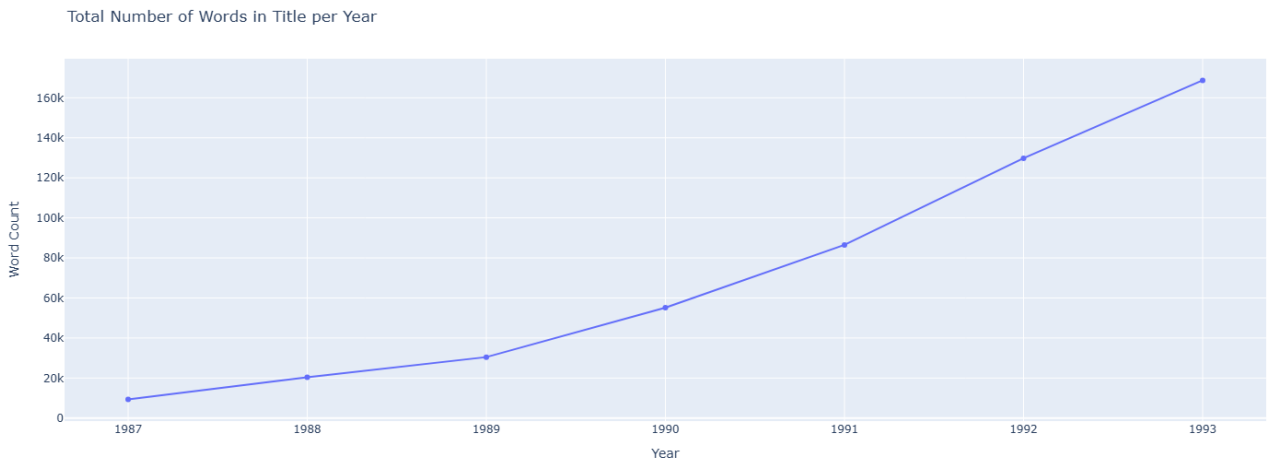
Εικόνα 4.20: Γράφημα 10 δημοφιλέστερων λέξεων στο title με 4 γράμματα και πάνω

Στο γράφημα της Εικόνας 4.20 η κορυφαία λέξη με περίπου 12.000 εμφανίσεις είναι η method που υποδεικνύει ότι οι περισσότερες πατέντες περιγράφουν κάποια μεθοδολογία. Οι επόμενες λέξεις στην κατάταξη είναι οι apparatus και process οι οποίες είναι ενδεικτικές βιομηχανικών, μηχανικών και επιστημονικών καινοτομιών. Ακολουθούν οι λέξεις device και system αντικατοπτρίζοντας την έμφαση σε κατασκευές και δομικά σχέδια. Τέλος οι λέξεις compositions, composition, preparation και containing δείχνουν ότι ένα μεγάλο μέρος των πατεντών σχετίζεται με τον κλάδο της Χημείας.

```
[ ]  date = pd.to_datetime(df[exanined_analytic_to_compare], format="%Y%m%d")
     df['year'] = date.dt.year

[ ]  def extract_all_words(text):
         return re.findall(r'\b[A-Za-z]+\b', str(text).lower())

[ ]  df['words'] = df[examined_analytic].apply(extract_all_words)

     df_words = (
         df[['year','words']]
           .dropna(subset=['year'])
           .explode('words')
           .rename(columns={'words':'word'})
     )

     words_per_year = (
         df_words
           .groupby('year')
           .size()
           .reset_index(name='word_count')
           .sort_values('year')
     )

     title = f"Total Number of Words in {examined_analytic_label} per Year"

     fig = px.line(
         words_per_year,
         x='year',
         y='word_count',
         title=title,
         labels={'word_count':'Word Count','year':'Year'}
     )

     fig.update_traces(mode='markers+lines')
     fig.update_layout(margin=dict(l=40, r=40, t=80, b=40))
     fig.show()
```

Εικόνα 4.21: Κώδικας γραφήματος πλήθους λέξεων ανά έτος

Για την απεικόνιση του πλήθους των λέξεων ανά έτος το πρώτο βήμα είναι η εξαγωγή του έτους από την στήλη date αφού έχει μετατραπεί σε αντικείμενο datetime. Έπεται η εφαρμογή της συνάρτησης extract_all_words σε όλες τις εγγραφές της στήλης title και η αποθήκευση στην νέα στήλη words στο dataframe. Στη συνέχεια δημιουργείται η μεταβλητή df_words από τις στήλες year και words, αφαιρώντας κενές τιμές από τα έτη και εφαρμόζοντας την συνάρτηση explode στην στήλη words. Η μεταβλητή words_per_year που χρησιμοποιείται για την δημιουργία του γραφήματος, φτιάχνεται κατηγοριοποιώντας τις εγγραφές των λέξεων με βάση το έτος και στη συνέχεια ταξινομώντας με βάση το έτος. Το γράφημα δημιουργήθηκε με την βιβλιοθήκη plotly και την συνάρτηση line η οποία δίνει την δυνατότητα δημιουργίας polyline γραφημάτων.



Εικόνα 4.22: Γράφημα πλήθους λέξεων ανά έτος για title

Από το γράφημα φαίνεται ξεκάθαρα σταθερή αύξηση στο πλήθος των λέξεων στους τίτλους μέσα στο χρόνο. Αυτή η αύξηση θα μπορούσε να οφείλεται στην αύξηση των λέξεων που χρησιμοποιούνται στους τίτλους αλλά αυτό μπορεί να καταρριφθεί εύκολα από το γράφημα της Εικόνας 4.4 που δείχνει ξεκάθαρα ότι η αύξηση οφείλεται στην αύξηση συνολικού πλήθους των πατεντών κάθε έτος.

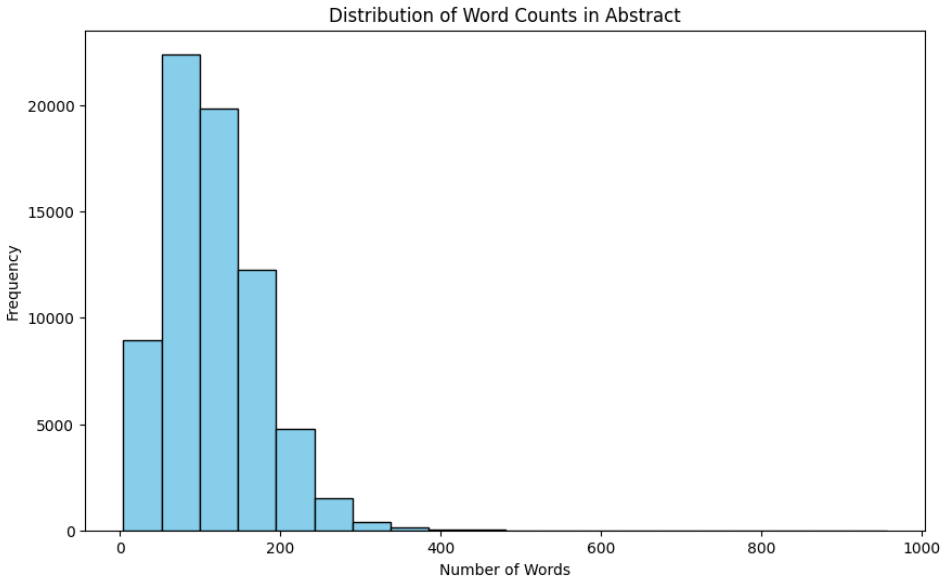
4.1.5 Ανάλυση Section Abstract

Το πεδίο που αναλύθηκε είναι το abstract, το οποίο αναφέρεται στην περίληψη κάθε πατέντας. Αφού φορτώθηκαν τα δεδομένα, ορίστηκε η μεταβλητή examined_analytic σε title και η examined_analytic_label σε Title. Χρησιμοποιώντας τον ίδιο κώδικα που χρησιμοποιήθηκε και για το title, όπως φαίνεται στην Εικόνα 4.18, υπολογίσθηκαν ο μέσος όρος και ο διάμεσος που είναι 115.5 και 108 αντίστοιχα.

```
title = f"Distribution of Word Counts in {examined_analytic_label}"

plt.figure(figsize=(10, 6))
plt.hist(df["word_count"], bins=20, color="skyblue", edgecolor="black")
plt.title(title)
plt.xlabel("Number of Words")
plt.ylabel("Frequency")
plt.show()
```

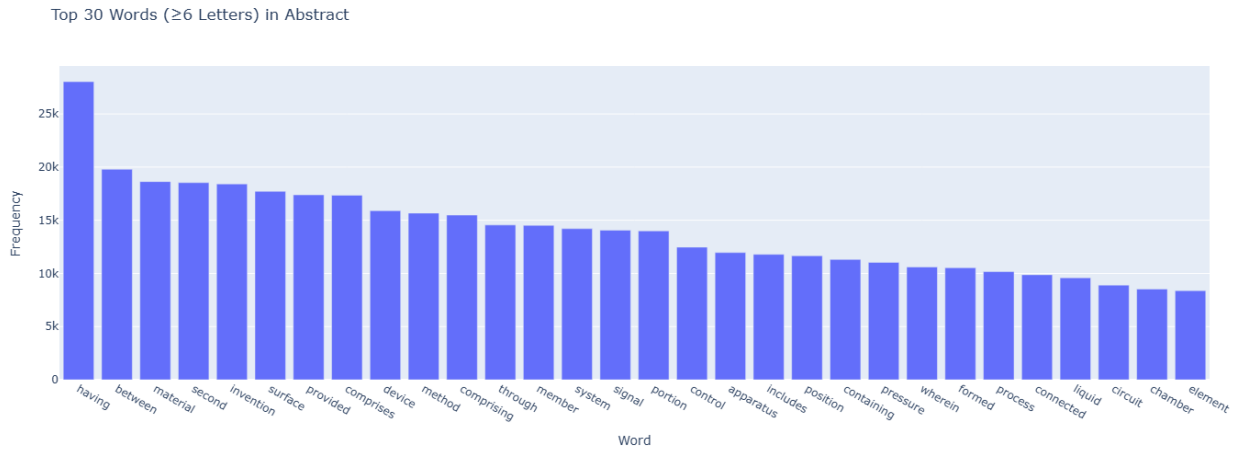
Εικόνα 4.23: Κώδικας κατανομής λέξεων για abstract

Στην Εικόνα 4.23 φαίνεται ο κώδικας για την δημιουργία ενός ιστογράμματος με την κατανομή των λέξεων του abstract. Η μεταβλητή df[‘word_count’], που φτιάχνεται όπως φαίνεται στην Εικόνα 3.18, χρησιμοποιείται σαν παράμετρος στην συνάρτηση hist της βιβλιοθήκης matplotlib. Από το παραγόμενο ιστόγραμμα που φαίνεται στην Εικόνα 4.24, είναι ξεκάθαρο πως η μεγάλη πλειοψηφία έχει πλήθος λέξεων στο εύρος 0-200, με την μεγαλύτερη συχνότητα να συγκεντρώνεται στο εύρος 50-150 υποδεικνύοντας ότι τα περισσότερα abstract είναι γραμμένα σε αυτό το εύρος λέξεων. Μετά τις 200 λέξεις παρουσιάζεται απότομη μείωση που συνιστά ότι αυτά τα πλήθη λέξεων είναι σχετικά σπάνια. Ακόμα, φαίνεται πως υπάρχουν κάποια μεγάλα abstract που φτάνουν έως και 500 λέξεις αποτελώντας ξεκάθαρες εξαιρέσεις στον κανόνα.



Εικόνα 4.24: Ιστόγραμμα κατανομής λέξεων του abstract

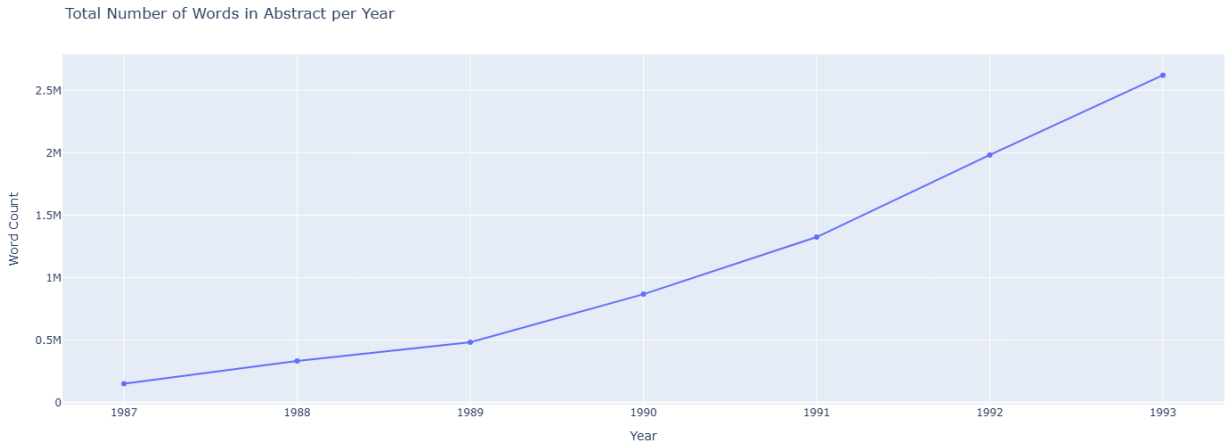
Για την εμφάνιση των 10 δημοφιλέστερων λέξεων, με πάνω από 4 γράμματα για να αποφευχθούν λέξεις που δεν προσφέρουν αξία στην ανάλυση, εκτελέστηκε ο ίδιος κώδικας που υλοποιήθηκε και για το title όπως φαίνεται στην Εικόνα 4.19. Το αποτέλεσμα σε αυτή την περίπτωση δεν είχε κανένα ουσιώδες αποτέλεσμα αφού το 80% των λέξεων ήταν αντωνυμίες όπως which, with, from, that, each, such κ.α. Εκτελώντας ξανά τον κώδικα για πλήθος γραμμάτων μεγαλύτερο η ίσο του 5 το αποτέλεσμα δεν ήταν πολύ διαφορετικό με τις μόνες λέξεις που προσφέρουν κάποια αξία στην ανάλυση να είναι η λέξη material στην θέση 7 με 18.634 εμφανίσεις, η λέξη invention στην θέση 9 με 18.417 εμφανίσεις και στην θέση 10 η λέξη surface με 17.722 εμφανίσεις. Παρόμοια ήταν τα αποτελέσματα και για το πλήθος των 6 λέξεων. Ένα συμπέρασμα που μπορεί να βγει είναι ότι όσο μεγαλώνει ο μέσος όρος των λέξεων σε ένα σύνολο κειμένων τόσο οι πιο χρησιμοποιούμενες λέξεις, τουλάχιστον οι πρώτες 10, θα είναι αντωνυμίες, επιρρήματα και γενικότερα λέξεις που εξυπηρετούν την γραμματική. Για το πλήθος των 6 γραμμάτων και πάνω, επιλέγοντας τις πρώτες 30 λέξεις, υπάρχουν κάποια δεδομένα που χρήζουν ερμηνείας.



Εικόνα 4.25: Γράφημα 30 δημοφιλέστερων λέξεων στο abstract με 6 γράμματα και πάνω

Όπως φαίνεται στην Εικόνα 4.25, λέξεις όπως "material", "invention", "device", "method", "comprising", "system", "signal", "apparatus" και "member" αντικατοπτρίζουν την ισχυρή τεχνική φύση

του περιεχομένου των abstract. Επίσης, λέξεις όπως "chamber", "circuit", "liquid", "pressure" και "element" υποδεικνύουν συγκεκριμένα τεχνικά πεδία όπως η μηχανική υλικών, η ηλεκτρονική και ίσως η δυναμική των ρευστών.



Εικόνα 4.26: Γράφημα πλήθους λέξεων ανά έτος για abstract

Το γράφημα που φαίνεται στην Εικόνα 4.26 δημιουργήθηκε εκτελώντας τον κώδικα που φαίνεται στην Εικόνα 4.21. Όπως και στην περίπτωση του title, φαίνεται σταθερή αύξηση στο πλήθος των λέξεων μέσα στο χρόνο, το οποίο είναι γνωστό ότι οφείλεται στην αύξηση του αριθμού των πατεντών μέσα στον χρόνο. Αν δεν υπήρχε αυτή η πληροφορία, ένας άλλος τρόπος για την εξαγωγή ασφαλούς συμπεράσματος θα ήταν ο υπολογισμός του μέσου όρου των λέξεων για κάθε έτος όπως φαίνεται στην Εικόνα 4.27. Είναι ξεκάθαρο λοιπόν πως εφόσον ο μέσος όρος των λέξεων έχει ελάχιστες μεταβολές μέσα στον χρόνο, η αύξηση των συνολικών λέξεων κάθε χρόνο οφείλεται στην αύξηση του αριθμού των πατεντών.

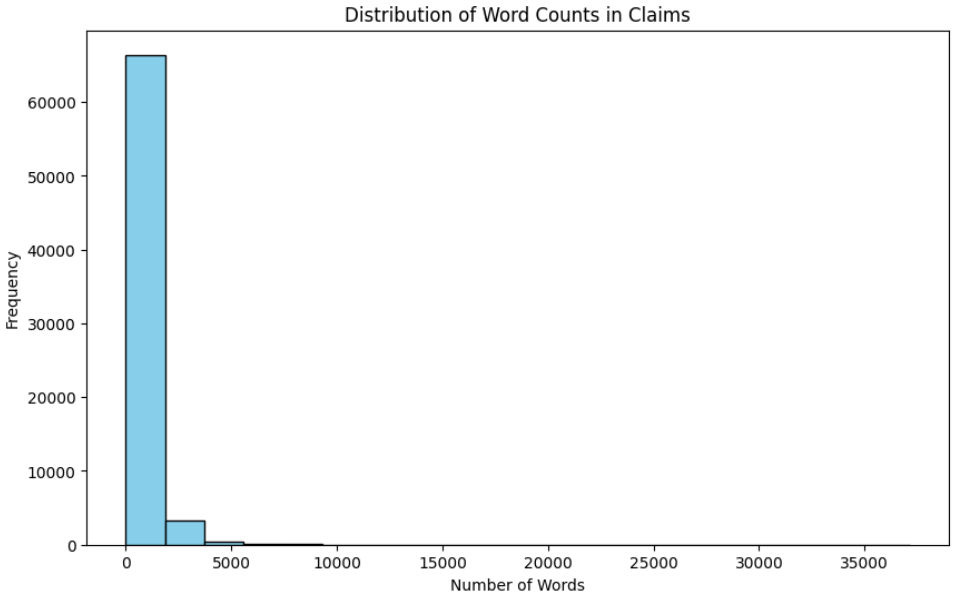
```
mean_words_per_year = df.groupby("year")["word_count"].mean().reset_index()

print(mean_words_per_year)

   year  word_count
0  1987  118.387556
1  1988  121.467909
2  1989  118.886268
3  1990  117.900026
4  1991  113.479506
5  1992  114.068979
6  1993  115.521088
```

Εικόνα 4.27: Μέσος όρος λέξεων abstract ανά έτος

4.1.6 Ανάλυση Section Claims

Το πεδίο που αναλύθηκε είναι το claims, που αναφέρεται στο νομικά δεσμευτικό τμήμα μιας πατέντας ή αλλιώς τις αξιώσεις. Αφού φορτώθηκαν τα δεδομένα ορίστηκαν κατάλληλα οι μεταβλητές examined_analytic και χρησιμοποιώντας τον κώδικα που χρησιμοποιήθηκε για το title και για το abstract, όπως φαίνεται στην Εικόνα 4.18 υπολογίστηκαν ο μέσος όρος και ο διάμεσος που είναι 760.4 και 561 αντίστοιχα. Για την δημιουργία ιστογράμματος με την κατανομή των λέξεων των claims,

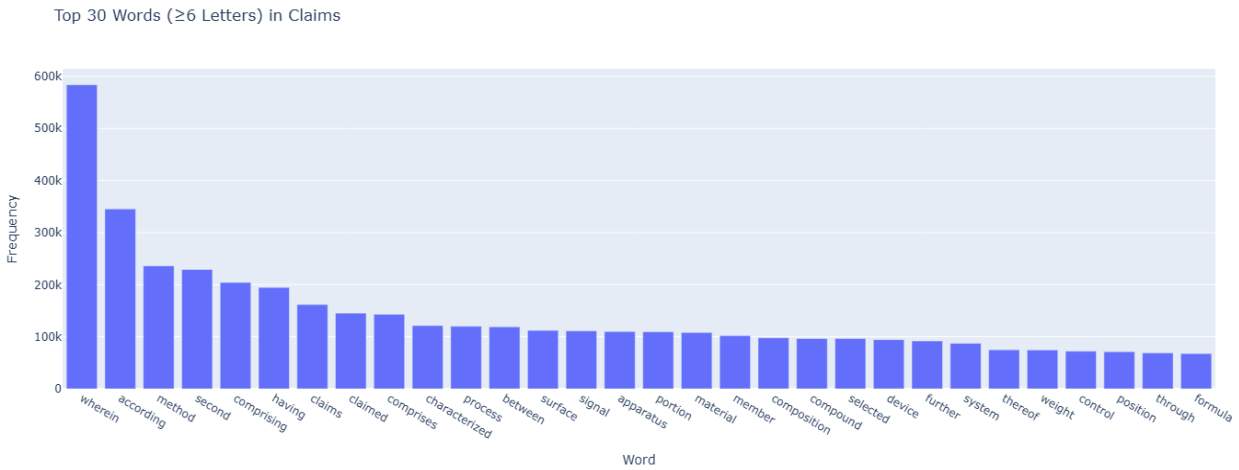
χρησιμοποιήθηκε ο αντίστοιχος κώδικας για το ιστόγραμμα του πεδίου abstract, όπως φαίνεται στην Εικόνα 3.23, ενώ η μεταβλητή df[‘word_count’] φτιάχνεται όπως φαίνεται στην Εικόνα 4.18.



Εικόνα 4.28: Ιστόγραμμα κατανομής του claims

Στο γράφημα της Εικόνας 4.28 φαίνεται ότι η συντριπτική πλειοψηφία έχει πλήθος λέξεων στο εύρος 0-2000, με την μεγαλύτερη συχνότητα να συγκεντρώνεται στο εύρος 0-500 υποδεικνύοντας ότι τα περισσότερα claims είναι γραμμένα σε αυτό το εύρος λέξεων. Η συχνότητα μειώνεται δραματικά περίπου μετά τις 3500 λέξεις, με κάποιες ακραίες τιμές να φτάνουν κοντά στις 7500 λέξεις.

Για την εμφάνιση των 10 δημοφιλέστερων λέξεων ακολουθήθηκε η ίδια μεθοδολογία με αυτήν για το abstract, όπου ο κώδικας εκτελέστηκε για πλήθος μεγαλύτερο ή ίσο των τεσσάρων (4), μεγαλύτερο ή ίσο των πέντε (5) και μεγαλύτερο ή ίσο των έξι (6) γραμμάτων. Όπως και στην περίπτωση του title, αλλά στην περίπτωση του πεδίου claims ακόμα περισσότερο αφού κατά μέσο όρο περιέχει περισσότερες λέξεις, είναι δύσκολο να βρεθούν λέξεις ανάμεσα στις δημοφιλέστερες, οι οποίες προσφέρουν κάποια αναλυτική αξία. Παρακάτω στην Εικόνα 4.29 φαίνεται το γράφημα για τις 30 δημοφιλέστερες λέξεις με 6 η περισσότερα γράμματα στα claims.



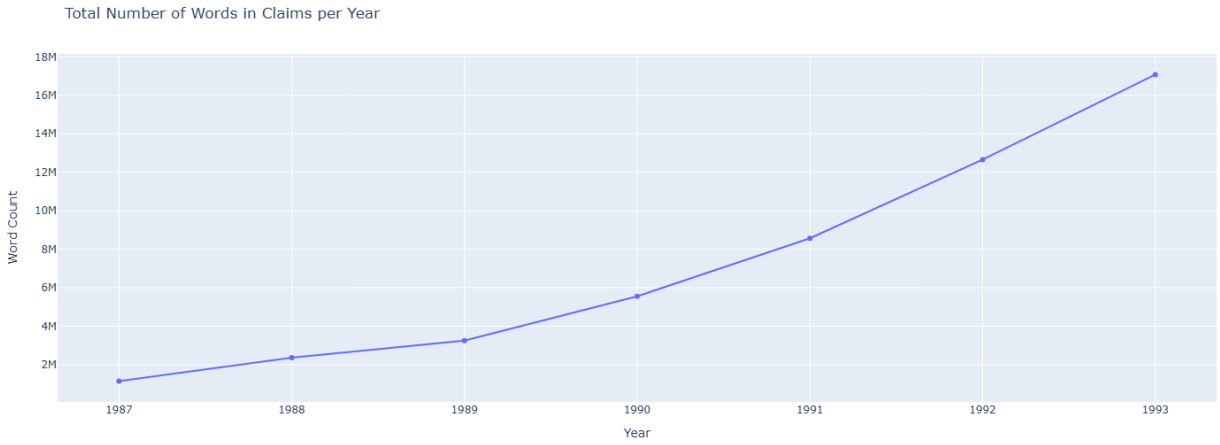
Εικόνα 4.29: Γράφημα 30 δημοφιλέστερων λέξεων στα claims με 6 γράμματα και πάνω

Για την δημιουργία του γραφήματος με τον συνολικό αριθμό λέξεων στα claims ανά έτος, παραμετροποιήθηκε ο κώδικας που χρησιμοποιήθηκε για το abstract, καθώς το αυξημένο πλήθος λέξεων στο πεδίο δημιούργησε την ανάγκη για αποδοτικότερη διαχείριση της μνήμης.

```
[ ]  df['word_count'] = df[examined_analytic].str.count(r'\b[A-Za-z]+\b')

     words_per_year = (
         df.groupby('year', as_index=False)['word_count']
           .sum()
     )
```

Εικόνα 4.30: Κώδικας για γράφημα συνολικού αριθμού λέξεων στα claims ανά έτος

Αντί να χρησιμοποιηθεί η μέθοδος apply όπως στις προηγούμενες περιπτώσεις, χρησιμοποιήθηκε η str.count για να μετρηθούν οι λέξεις με regular expression, η οποία είναι υλοποιημένη σε C και εκτελείται μια φορά πάνω σε ολόκληρο το σύνολο των επιθυμητών γραμμών και όχι διατρέχοντας τες. Επίσης δεν χρησιμοποιήθηκε η μέθοδος explode και έτσι δεν χρειάστηκε να δημιουργηθεί μια καινούργια γραμμή για κάθε λέξη αλλά ένας integer αριθμός με το πλήθος των λέξεων για κάθε πατέντα. Με τον τρόπο αυτό αξιοποιήθηκε ελάχιστη μνήμη RAM για την διαχείριση ενός τεράστιου όγκου δεδομένων, στην συγκεκριμένη περίπτωση εκατομμύρια λέξεων όπως φαίνεται και στο γράφημα της Εικόνας 4.31



Εικόνα 4.31: : Γράφημα πλήθους λέξεων ανά έτος για claims

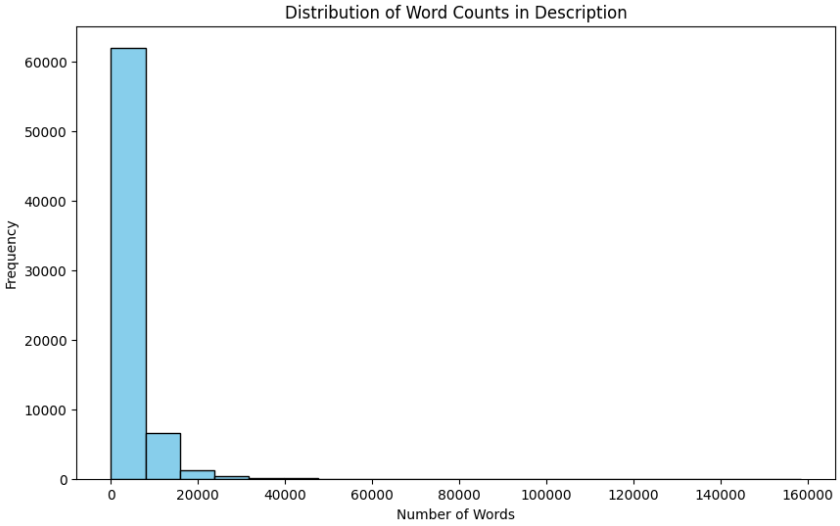
Όπως και στην περίπτωση του title και του abstract, φαίνεται σταθερή αύξηση στο πλήθος των λέξεων μέσα στο χρόνο, το οποίο είναι γνωστό ότι οφείλεται στην αύξηση του πλήθους των πατεντών μέσα στον χρόνο. Χρησιμοποιώντας τον αντίστοιχο κώδικα όπως και στην περίπτωση του πεδίου abstract, υπολογίσθηκε ο μέσος όρος των λέξεων ανά έτος ο οποίος φαίνεται στην Εικόνα 4.32. Παρότι τα πρώτα τρία χρόνια ο μέσος όρος είναι υψηλότερος σε σχέση με τα επόμενα, ο ραγδαίος ρυθμός αύξησης του πλήθους των πατεντών υπερκαλύπτει αυτή την μείωση με αποτέλεσμα το συνολικό πλήθος των λέξεων να αυξάνεται μέσα στον χρόνο.

	year	word_count
0	1987	854.524381
1	1988	814.650104
2	1989	759.024521
3	1990	717.387934
4	1991	700.972184
5	1992	696.705147
6	1993	720.322311

Εικόνα 4.32: Μέσος όρος λέξεων στα claims ανά έτος

4.1.7 Ανάλυση Section Description

Το πεδίο που αναλύθηκε είναι το description, που αναφέρεται στην λεπτομερή τεχνική παρουσίαση της εφεύρεσης. Όπως και για τα προηγούμενα λεκτικά πεδία, υπολογίσθηκε ο μέσος όρος και ο διάμεσος που είναι 4562.7 και 3359 αντίστοιχα αλλά και το ιστόγραμμα της κατανομής του πλήθους των λέξεων στο description.



Εικόνα 4.33: Ιστόγραμμα κατανομής πλήθους λέξεων στο description

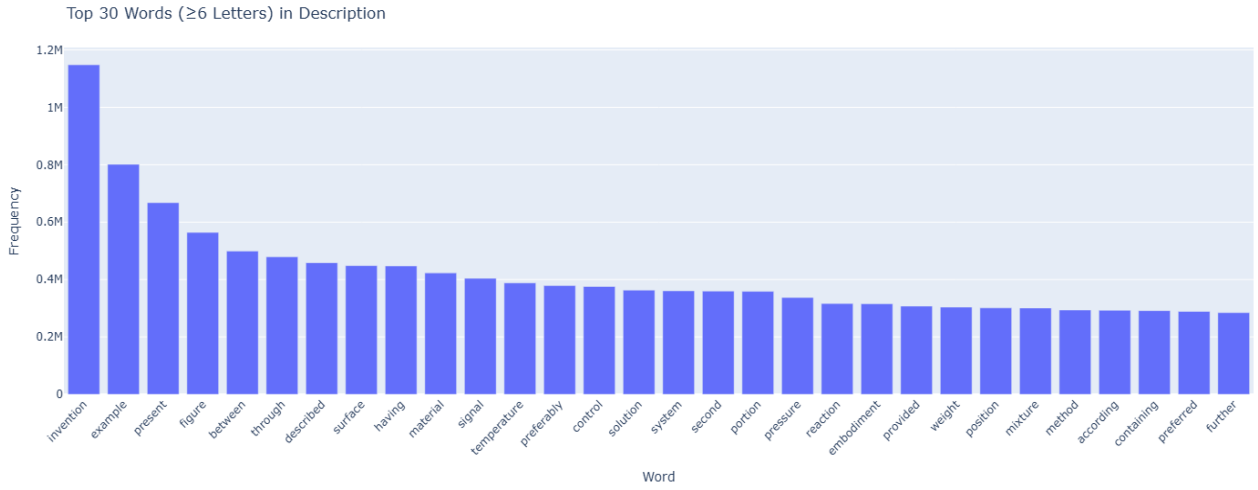
Όπως και τα ιστογράμματα των κατανομών των υπόλοιπων λεκτικών πεδίων, η μεγάλη πλειοψηφία έχει πλήθος σε συγκεκριμένο εύρος, στην συγκεκριμένη περίπτωση 0-10000 περίπου, την μεγαλύτερη συχνότητα στο εύρος 0-5000 ενώ υπάρχουν και κάποιες περιπτώσεις που ξεφεύγουν πάνω από τις 40000 χιλιάδες λέξεις. Για τον υπολογισμό των δημοφιλέστερων λέξεων δημιουργήθηκε η ανάγκη για παραμετροποίηση του κώδικα, καθώς όπως φαίνεται και από το γράφημα της Εικόνας 4.29, το πεδίο έχει μεγάλο όγκο στοιχείων προς επεξεργασία ανά γραμμή και άρα υψηλές απαιτήσεις από υπολογιστική ισχύ.

```
texts = df[examined_analytic].dropna().astype(str)

letters = 6
n = 30

vectorizer = CountVectorizer(
    token_pattern=rf"[A-Za-z]{{{letters},}}",
    lowercase=True,
    max_features=n
)

X = vectorizer.fit_transform(texts)
counts = X.sum(axis=0).A1
words = vectorizer.get_feature_names_out()
```

```
[ ] top_df = pd.DataFrame({
        'word': words,
        'count': counts
    }).sort_values('count', ascending=False)
```

Εικόνα 4.34: Χρήση μεθόδου CountVectorizer για υπολογισμό δημοφιλέστερων λέξεων στο description

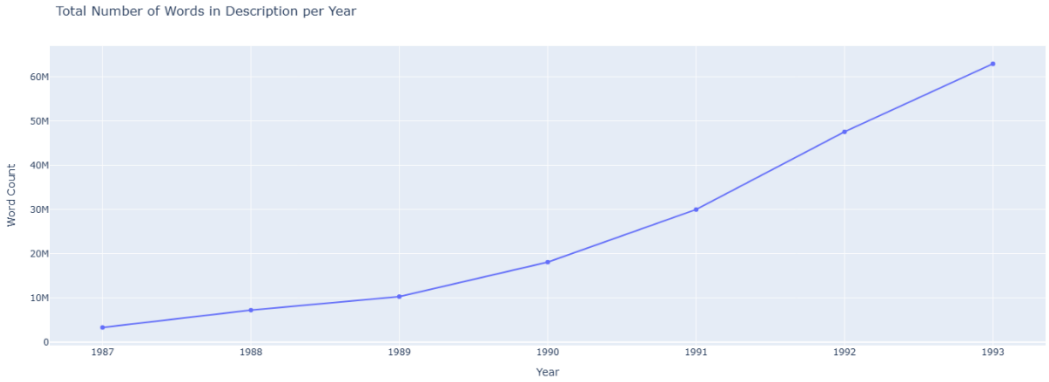
Όπως φαίνεται στην Εικόνα 4.34 χρησιμοποιήθηκε η μέθοδος CountVectorizer της βιβλιοθήκης scikit-learn [26] η οποία μετατρέπει μια συλλογή δεδομένων κειμένου σε ένα πίνακα που περιέχει τα πλήθη των tokens, το σύνολο των χαρακτήρων που έχουν σημασιολογικό νόημα. Οι κύριες λειτουργικότητες της μεθόδου, δηλαδή το tokenization (δημιουργία των tokens) και το μέτρημα των tokens, είναι υλοποιημένες σε μεταγλωττισμένο κώδικα της γλώσσας C, οποίος είναι πολύ πιο γρήγορος. Επίσης η μέθοδος για την αποθήκευση των tokens, κατασκευάζει ένα αραιό πίνακα ο οποίος περιέχει μόνο μη μηδενικά πλήθη, άρα αν το κείμενο έχει τεράστιο λεξιλόγιο αλλά δεν χρησιμοποιούνται όλα τα tokens του κειμένου τότε εξοικονομείται πάρα πολλή μνήμη RAM. Στην περίπτωση του description επιλέχθηκαν σαν tokens μόνο οι λέξεις με 6 γράμματα και πάνω χρησιμοποιώντας το κατάλληλο regex στην παράμετρο token_pattern της μεθόδου. Με την fit_transform δημιουργείται ο αραιός πίνακας και αφού γίνουν οι απαραίτητοι μετασχηματισμοί δημιουργείται το dataframe top_df το οποίο χρησιμοποιείται για την δημιουργία του γραφήματος με την βιβλιοθήκη plotly.



Εικόνα 4.35: Γράφημα 30 δημοφιλέστερων λέξεων στο description με 6 γράμματα και πάνω

Για την δημιουργία του γραφήματος του πλήθους λέξεων ανά έτος χρησιμοποιήθηκε ο ίδιος κώδικας που υλοποιήθηκε για το πεδίο claims με την χρήση της μεθόδου str.count ώστε να αξιοποιηθεί όσο το

δυνατόν λιγότερη μνήμη RAM για την διαχείριση του τεράστιου όγκου δεδομένων. Το γράφημα όπως φαίνεται και στην Εικόνα 4.36, δεν διαφέρει σχεδόν καθόλου από τα αντίστοιχα για τα υπόλοιπα λεκτικά πεδία. Η μόνη διαφορά είναι η κλίμακα του πλήθους των λέξεων, με την πρώτη τιμή να είναι τα 3.5 εκατομμύρια και η τελευταία τα 62 εκατομμύρια. Όπως και για τα υπόλοιπα λεκτικά πεδία η ραγδαία αυτή αύξηση οφείλεται στην αύξηση του αριθμού των πατεντών, το οποίο επιβεβαιώνεται και από τον μέσο όρο των λέξεων ανά έτος στο description ο οποίος κυμαίνεται από τις 2477 λέξεις μέχρι τις 2653.



Εικόνα 4.36: Γράφημα πλήθους λέξεων ανά έτος για description

4.2 Επίλογος

Στο Κεφάλαιο αυτό παρουσιάστηκαν οι αναλύσεις για την ημερομηνία δημοσίευσης, τα ονόματα των εφευρετών, τους main κωδικούς, τον τίτλο, την περίληψη, τις αξιώσεις και την περιγραφή κάθε πατέντας. Εξηγήθηκε ο κώδικας για την εξαγωγή αναλυτικών δεδομένων από τα πεδία ενώ τα παραγόμενα γραφήματα τους αξιοποιήθηκαν ώστε να προκύψουν χρήσιμα συμπεράσματα.

Κεφάλαιο 5ο: Μεταφορά ανάλυσης στη συλλογή WPI

Στο κεφάλαιο αυτό περιλαμβάνεται η μεταφορά της μεθοδολογίας της ανάλυσης του Κεφαλαίου 4 σε ένα νέο εξειδικευμένο υποσύνολο της συλλογής WPI, η αξιολόγηση της μεταφοράς και η σύγκριση των αποτελεσμάτων της ανάλυσης με τα αντίστοιχα αποτελέσματα του dataset CLEF-IP-2011_EN_All_MainClass.

5.1 Δημιουργία νέου υποσυνόλου της συλλογής WPI

Η δημιουργία του νέου υποσυνόλου της WPI έγινε με δύο κριτήρια. Το πρώτο είναι η επιλογή των δεδομένων. Πιο συγκεκριμένα, αποφασίστηκε ο αριθμός των πατεντών που θα επιλέγεται από κάθε μήνα να είναι ανάλογος του συνολικού αριθμού των πατεντών για τον εκάστοτε μήνα ενώ η ίδια συνθήκη πρέπει να ισχύει και για το πεδίο kind, δηλαδή ο αριθμός από κάθε kind που επιλέγεται για κάθε μήνα να είναι ανάλογος των συνολικών πατεντών με αυτό το kind για τον εκάστοτε μήνα. Για παράδειγμα, αν οι πατέντες με kind A1 για τον μήνα Ιούνιο αποτελούν το 10% του συνόλου των δεδομένων, θα επιλεχθούν τα ανάλογα δεδομένα έτσι ώστε να αποτελούν και το 10 % του υποσυνόλου. Το δεύτερο κριτήριο είναι η δομή του υποσυνόλου η οποία διαμορφώθηκε με τέτοιο τρόπο που να επιτρέπει την εφαρμογή του κώδικα που αναλύθηκε στο Κεφάλαιο 4 για την εξαγωγή αναλυτικών στοιχείων. Στο πιο αφηρημένο επίπεδο, πρόκειται για μια πινακοειδή μορφή η οποία αποτελείται από στήλες οι οποίες δηλώνουν το εκάστοτε πεδίο και γραμμές οι οποίες περιέχουν την τιμή για το κάθε πεδίο. Κάθε στοιχείο, από μια συγκεκριμένη προεπιλογή στοιχείων, εξάχθηκε από τα πρωτογενή δεδομένα σε μορφή XML της WPI και δημιουργήθηκε μια εγγραφή σε ένα CSV αρχείο. Όσα στοιχεία της WPI υπάρχουν και σαν πεδία στο dataset CLEF-IP-2011_EN_All_MainClass ενσωματώθηκαν στο υποσύνολο ενώ τα text πεδία (abstract, description, claims) δεν ενσωματώθηκαν αυτούσια αλλά ένα boolean πεδίο που υποδηλώνει την ύπαρξη ή μη του πεδίου. Πρέπει να σημειωθεί ότι το υποσύνολο αποτελείται από 70000 εγγραφές και δημιουργήθηκε μόνο από πατέντες της WPI με το πρότυπο EP. Στην Εικόνα 5.1 φαίνονται τα πεδία του υποσυνόλου

```
columns_order = [
    "ucid",
    "kind",
    "lang",
    "date",
    "main_code",
    "further_codes",
    "ipcr_codes",
    "cpc_codes",
    "abstract_en",
    "description_en",
    "claims_en",
    "abstract_fr",
    "description_fr",
    "claims_fr",
    "abstract_de",
    "description_de",
    "claims_de",
]
```

Εικόνα 5.1: : Πεδία εξειδικευμένου υποσυνόλου WPI

Με το κομμάτι κώδικα που φαίνεται στην Εικόνα 5.2 διατρέχεται αναδρομικά η φακελική δομή της WPI και αποθηκεύεται στην μεταβλητή files_by_month_kind οι διαδρομές των αρχείων των πατεντών για κάθε ζευγάρι Μήνα (σε 8ψήφιο format) – Kind

```
# Traverse the root folder
# Expecting top-level folders to be 8-digit publication dates (e.g., "20140101")
for date_folder in os.listdir(root_folder):
    if date_folder.isdigit() and len(date_folder) == 8:
        # Extract month as the first 6 digits (YYYYMM)
        month = date_folder[:6]
        date_path = os.path.join(root_folder, date_folder)
        if os.path.isdir(date_path):
            # Within each date folder, expect subfolders for kind codes (e.g., "A1", "B1")
            for kind_folder in os.listdir(date_path):
                kind_path = os.path.join(date_path, kind_folder)
                if os.path.isdir(kind_path):
                    # Recursively collect all XML files in this kind folder
                    for dirpath, _, filenames in os.walk(kind_path):
                        for filename in filenames:
                            if filename.lower().endswith(".xml"):
                                file_path = os.path.join(dirpath, filename)
                                files_by_month_kind[(month, kind_folder)].append(
                                    file_path
                                )
```

Εικόνα 5.2: Κώδικας προσπέλασης φακελικής δομής WPI

Οι φάκελοι πρώτου επιπέδου πρέπει να έχουν όνομα 8 ψηφίων (ημερομηνία) ενώ οι φάκελοι δεύτερου επιπέδου είναι kind codes. Στο κομμάτι κώδικα της Εικόνας 5.3 φαίνεται η μεθοδολογία της αναλογικής επιλογής των πατεντών του υποσυνόλου.

```
# For each group (month, kind), compute the sample count proportionally
for (month, kind), files in files_by_month_kind.items():
    count = len(files)
    sample_count = round((count / total_files) * target_subset)
    # Ensure we don't oversample
    sample_count = min(sample_count, len(files))
    sampled_files = random.sample(files, sample_count)

    for file_path in sampled_files:
        counter += 1
        if counter % 1000 == 0:
            print(f"Processed {counter} patents so far")

        parsed_data = parse_patent_xml(file_path)
        if parsed_data:
            parsed_data["Month"] = month
            parsed_data["Kind"] = kind
            patent_data.append(parsed_data)
```

Εικόνα 5.3: Κώδικας μεθοδολογίας αναλογικής επιλογής πατεντών υποσυνόλου WPI

Η γραμμή for (month, kind), files in files_by_month_kind.items(): διατρέχει κάθε ζευγάρι κλειδιού-τιμής στο λεξικό files_by_month_kind. Κάθε κλειδί είναι ένα tuple (month, kind) , π.χ. ('201401', 'A1') ενώ κάθε files είναι μια λίστα με τις διαδρομές των αρχείων των XML σε αυτό το group. Έπειτα με την γραμμή count = len(files) μετριούνται πόσα αρχεία XML υπάρχουν σε αυτό το group. Στην γραμμή sample_count = round((count / total_files) * target_subset) γίνεται ο υπολογισμός του πόσα αρχεία να επιλεχθούν από αυτό το group. Είναι αναλογικό, αν αυτό το group αντιπροσωπεύει το 10% όλων των αρχείων, τότε επιλέγεται το 10% του επιθυμητού δείγματος (π.χ. 2.000 από 20.000). Το total_files είναι ο συνολικός αριθμός όλων των αρχείων XML. Στη συνέχεια με την γραμμή sample_count = min(sample_count, len(files)) εξασφαλίζεται ότι ο αριθμός των δειγμάτων δεν ξεπερνά τον αριθμό των διαθέσιμων αρχείων. Στην γραμμή sampled_files = random.sample(files, sample_count) γίνεται η τυχαία επιλογή του δείγματος των αρχείων από το group. Στην γραμμή parsed_data = parse_patent_xml(file_path) γίνεται το parse των επιθυμητών στοιχείων μέσα από το XML καλώντας την συνάρτηση parse_patent_xml. Το parse γίνεται με την χρήση της βιβλιοθήκης BeautifulSoup. Ενδεικτικά παρουσιάζεται το parse των στοιχείων ucid και date στην Εικόνα 5.4

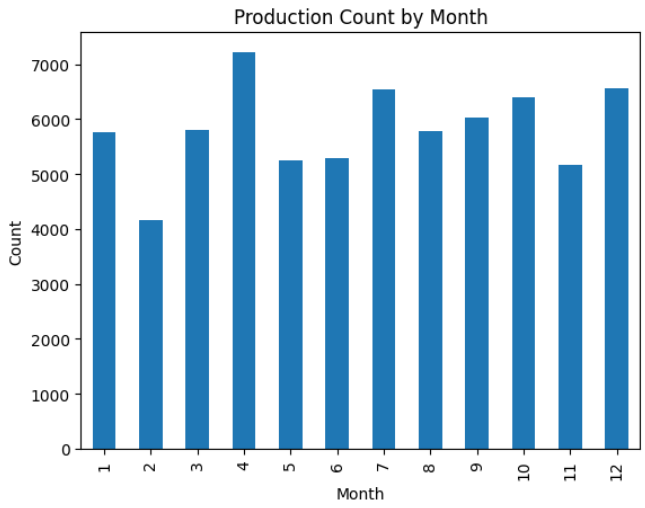
```
soup = BeautifulSoup(content, "xml")
document_info = soup.find_all("patent-document")

ucid = (
    document_info[0]["ucid"]
    if document_info and "ucid" in document_info[0].attrs
    else "N/A"
)
date = (
    document_info[0]["date"]
    if document_info and "date" in document_info[0].attrs
    else "N/A"
)
```

Εικόνα 5.4: Parse στοιχείων ucid και date στοιχείων της WPI

Αφού διατρεχθούν όλα τα groups και γίνει η εξαγωγή όλων των στοιχείων, αποθηκεύονται σε ένα αρχείο CSV. Για την αποθήκευση και διανομή αυτού του υποσυνόλου δημιουργήθηκε ένα νέο dataset repository στο Hugging Face.

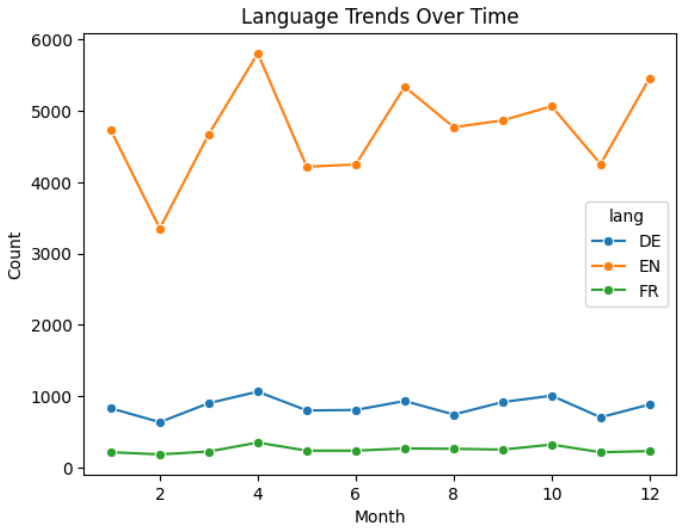
5.2 Ανάλυση δεδομένων υποσυνόλου WPI

Το πεδίο που επιλέχθηκε για την εξαγωγή αναλυτικών δεδομένων είναι το date, που αναφέρεται στην ημερομηνία δημοσίευσης της κάθε πατέντας. Τα δεδομένα φορτώθηκαν χρησιμοποιώντας την βασική μέθοδο του Hugging Face load_dataset και όχι το προσαρμοσμένο loading script που αναπτύχθηκε αφού το υποσύνολο προς ανάλυση αποτελείται από ένα CSV αρχείο. Κατά την φόρτωση ορίστηκαν οι τύποι των πεδίων, όπως και στο προσαρμοσμένο loading script, για αποδοτικότερη χρήση της μνήμης. Στη συνέχεια υπολογίστηκαν ο μέσος όρος, ο διάμεσος και η τυπική απόκλιση του έτους με τιμές 2014.5, 2015 και 0.49 αντίστοιχα χρησιμοποιώντας τον ίδιο κώδικα που χρησιμοποιήθηκε για τον υπολογισμό των αντίστοιχων στοιχείων για το dataset CLEF-IP-2011_EN_All_MainClass. Όπως είναι γνωστό η συλλογή WPI και κατ' επέκταση το εξειδικευμένο υποσύνολο περιέχει δεδομένα πατεντών από δύο έτη, τα 2014 και 2015. Για τον λόγο αυτό επιλέχθηκε οι αναλύσεις στο υποσύνολο να γίνουν με βάση τον μήνα και όχι το έτος. Για την δημιουργία του γραφήματος του πλήθους των πατεντών ανά μήνα δημοσίευσης χρησιμοποιήθηκε ο ίδιος κώδικας που αναπτύχθηκε για την δημιουργία του γραφήματος του πλήθους πατεντών ανά έτος δημοσίευσης για το dataset CLEF-IP-2011_EN_All_MainClass όπως φαίνεται στην Εικόνα 3.3. Το παραγόμενο γράφημα φαίνεται στην Εικόνα 5.5



Εικόνα 5.5: Γράφημα πλήθους πατεντών ανά μήνα δημοσίευσης

Όπως φαίνεται και στο γράφημα ο Απρίλιος είναι ο μήνας με το μεγαλύτερο πλήθος ενώ ο Φεβρουάριος αυτός με τις λιγότερες. Δύο μήνες που επίσης έχουν μεγαλύτερα πλήθη σε σχέση με τους υπόλοιπους είναι ο Ιούλιος και ο Δεκέμβριος, ενώ οι υπόλοιποι έχουν πανομοιότυπα πλήθη. Παρότι τα πλήθη στην ολότητα τους δεν εμφανίζουν μεγάλες αυξομειώσεις, οι διαφορές στις τιμές των πληθών μπορεί να οφείλονται σε εποχιακή ζήτηση, όπως προθεσμίες ενόψει καλοκαιριού τον Απρίλιο ή εντονότερη δραστηριότητα τον Δεκέμβριο λόγω ανάγκης επίτευξης ορισμένων στόχων ενόψει του τέλους του έτους. Το δεύτερο γράφημα που δημιουργήθηκε είναι το πλήθος των πατεντών ανά γλώσσα, ανά μήνα χρησιμοποιώντας τον ίδιο κώδικα για την δημιουργία του γραφήματος του πλήθους των πατεντών ανά γλώσσα, ανά έτος για το dataset CLEF-IP-2011_EN_All_MainClass όπως φαίνεται στην Εικόνα 4.5. Το παραγόμενο γράφημα φαίνεται στην Εικόνα 5.6



Εικόνα 5.6: Γράφημα πλήθους πατεντών ανά γλώσσα, ανά μήνα

Όπως φαίνεται στο γράφημα της Εικόνας 5.6 η αγγλική γλώσσα κυριαρχεί όλους τους μήνες με τις υψηλότερες τιμές των πληθών να ταυτίζονται με αυτές του γραφήματος της Εικόνας 5.5. Δεύτερη με αρκετά μεγάλη απόσταση είναι η γερμανική γλώσσα η οποία ωστόσο παρουσιάζει κάποιες αυξομειώσεις στα πλήθη, αντίθετα με την γαλλική η οποία έχει σχεδόν γραμμική πορεία μέσα στον χρόνο.

5.3 Αξιολόγηση μεταφερσιμότητας της μεθοδολογίας

Για την ανάλυση του πεδίου date του υποσυνόλου της WPI, εκτελέστηκε ο ίδιος ακριβώς κώδικας με αυτόν για το αντίστοιχο πεδίο για το dataset CLEF-IP-2011_EN_All_MainClass. Η μόνη διαφορά ήταν ότι η ανάλυση για την WPI έγινε σε επίπεδο μήνα ενώ για το CLEF-IP-2011_EN_All_MainClass σε επίπεδο έτους, ορίζοντας την μεταβλητή examined_sub_analytic σε month αντί για year. Αυτό βέβαια έγινε γιατί η κατανομή των δεδομένων του υποσυνόλου δεν πρόσφερε χρήσιμες αναλύσεις σε επίπεδο έτους και όχι γιατί δεν το επέτρεπε ο κώδικας. Το σημαντικό κομμάτι είναι η μορφή την οποία πρέπει να έχει το εκάστοτε dataset για να γίνει η εξαγωγή των αναλυτικών δεδομένων. Στην συγκεκριμένη περίπτωση, παρότι η δομή των πρωτογενών δεδομένων των συλλογών διαφέρει, ήταν εφικτό να εκτελεστεί ακριβώς ο ίδιος κώδικας για την δημιουργία analytics, αφού μετατράπηκαν σε μια πανομοιότυπη ενδιάμεση μορφή. Μια διαφορά που είχαν τα υποσύνολα ως προς την δομή τους ήταν ότι τα λεκτικά πεδία του υποσυνόλου WPI είχαν την μορφή boolean πεδίου και δεν περιείχαν το αυτούσιο κείμενο όπως στην περίπτωση του CLEF-IP-2011_EN_All_MainClass. Αυτό έγινε συνειδητά για να μην επιβαρυνθεί επιπλέον το υποσύνολο καθώς ήταν γνωστό πως δεν ήταν προς εξέταση τα πεδία

αυτά και όχι γιατί δεν ήταν εφικτό να ενσωματωθούν τα κείμενα αυτούσια. Το ίδιο ισχύει και για τα ονόματα των εφευρετών και των αιτούντων. Γενικότερα, γίνεται αντιληπτό πως οποιεσδήποτε πρωτογενείς συλλογές μπορούν αναλυθούν με την μεθοδολογία που αναπτύχθηκε, αρκεί να μετατραπούν πρώτα σε μια σαφώς ορισμένη κοινή, αν όχι ίδια μορφή. Είναι ξεκάθαρο πως λοιπόν πως η μεθοδολογία για την εξαγωγή των αναλύσεων είναι μεταφέρσιμη σε πολύ μεγάλο βαθμό, αν όχι πλήρως.

Κεφάλαιο 6ο: Συμπεράσματα και μελλοντικές βελτιώσεις

6.1 Συμπεράσματα

Μετά την θεωρητική ανάλυση των εργαλείων που χρησιμοποιήθηκαν και την υλοποίηση του πρακτικού μέρους προκύπτουν χρήσιμα συμπεράσματα. Αρχικά μέσω της διαδικασίας της επεξεργασίας των πρωτογενών δεδομένων της συλλογής CLEF-IP που περιέχει 3.5 εκατομμύρια έγγραφα πατεντών , της δημιουργίας του υποσυνόλου και του ανεβάσματος του στην πλατφόρμα Hugging Face, σε συνδυασμό με το προσαρμοσμένο loading script γίνεται προσβάσιμη στην κοινότητα μια πολύ μεγάλη συλλογή δεδομένων η οποία μπορεί να χρησιμοποιηθεί από τον εκάστοτε ερευνητή για αναπαραγώγιμη έρευνα στον τομέα της ανάκτησης και ταξινόμησης πατεντών. Επίσης η ύπαρξη του versioning συστήματος στα δεδομένα του υποσυνόλου παρέχει την δυνατότητα παρακολούθησης του ιστορικού των δεδομένων αλλά και την απρόσκοπτη εκτέλεση των ίδιων πειραμάτων σε διαφορετικές εκδόσεις του dataset.

Στο πλαίσιο της παρούσας εργασία έγινε εξαγωγή analytics πάνω σε κάποια πεδία του dataset που επιλέχθηκαν ως καταλληλότερα για εξαγωγή χρήσιμων συμπερασμάτων, χωρίς αυτό να αποκλείει την δυνατότητα ενσωμάτωσης και των υπόλοιπων πεδίων στα πειράματα. Ο κώδικας είναι παραμετροποιήσιμος και δομημένος με τέτοιο τρόπο που είναι δυνατό να εξαχθούν οι αναλύσεις παρόμοιων πεδίων αβίαστα, αλλάζοντας μόνο το όνομα του πεδίου προς ανάλυση. Για παράδειγμα η ανάλυση που έγινε για το πεδίο main_codes μπορεί να αναπαραχθεί για το πεδίο ecla_codes αλλάζοντας μια γραμμή κώδικα. Μέσω της εξαγωγής των αναλυτικών δεδομένων (analytics) που έγινε στο dataset γίνεται ξεκάθαρο πως ο μεγάλος αυτός όγκος δεδομένων μπορεί να αξιοποιηθεί για ποικίλες εφαρμογές όπως την αποκάλυψη προτύπων βιομηχανικής δραστηριότητας ή την κατανόηση των τεχνολογικών τάσεων και την αξιολόγηση της καινοτομίας.

Πέρα από την αναπαραγωγιμότητα σε επίπεδο πεδίων, η μεθοδολογία της εξαγωγής των analytics μπορεί να εφαρμοστεί αν όχι εξολοκλήρου, σε μεγάλο βαθμό σε άλλες συλλογές δεδομένων όπως αυτό έγινε αντιληπτό δημιουργώντας ένα καινούριο υποσύνολο της συλλογής WPI και εκτελώντας την εξαγωγή των αναλυτικών δεδομένων για ένα από τα πεδία της. Το σημαντικό κομμάτι σε αυτό το βήμα είναι ότι όσο και αν διαφέρουν τα πρωτογενή δεδομένα και η δομή των συλλογών είναι κρίσιμο να μετατραπούν σε μια κοινή ενδιάμεση μορφή η οποία λειτουργεί σαν σημείο αναφοράς για την εκτέλεση των πειραμάτων

6.2 Προτάσεις για μελλοντική βελτίωση

Όπως κάθε υλοποίηση και σύστημα επιδέχεται βελτίωση έτσι και εδώ, νέες προσθήκες και μεταβολές μπορούν να συμβάλλουν στην εξέλιξη των εργαλείων που παρουσιάστηκαν. Μία από αυτές θα μπορούσε να είναι η χρήση τεχνικών streaming όπως το Apache Spark για την παράλληλη επεξεργασία των XML αρχείων και την ταχύτερη κατασκευή των υποσυνόλων, μειώνοντας δραστικά το χρόνο εκτέλεσης σε περιβάλλοντα με πολλαπλούς επεξεργαστές. Η παράλληλη επεξεργασία των XML με αυτό το εργαλείο επιτυγχάνει δύο κρίσιμους στόχους: α) αποφυγή πλήρους φόρτωσης του αρχείου στη μνήμη (out-of-core processing) και β) αξιοποίηση όλων των διαθέσιμων πυρήνων ή κόμβων ενός cluster για κατανεμημένη επεξεργασία. Επίσης η διασύνδεση με Apache Spark μπορεί να κάνει αποδοτικότερη και την διαδικασία της εξαγωγής των αναλυτικών δεδομένων από τα υποσύνολα. Το Spark σπάει μεγάλα αρχεία σε partitions (π.χ. 128 MB blocks), στέλνει κάθε partition σε διαφορετικό executor, και εκτελεί τα μετασχηματιστικά βήματα (π.χ. έξαγωγή πεδίων, κανονικοποίηση, φίλτρα) παραλληλισμένα. Αυτό μειώνει τον χρόνο ολοκλήρωσης σχεδόν γραμμικά με τον αριθμό των διαθέσιμων

πυρήνων/κόμβων. Ακόμα με την χρήση του Lazy Evaluation, οι μετασχηματισμοί συσσωρεύονται και εκτελούνται μόνο όταν ζητηθεί ένα action (π.χ. write, count), βελτιστοποιώντας το πλάνο εκτέλεσης.

Μια ακόμα βελτίωση μπορεί να είναι η ενσωμάτωση κατανεμημένων NoSQL βάσεων δεδομένων όπως το Elasticsearch το οποίο μπορεί να αναβαθμίσει σημαντικά την ευελιξία και την απόδοση του συστήματος κατά την αποθήκευση και ανάκτηση των μεταδεδομένων των πατεντών σε πραγματικό χρόνο. Πιο συγκεκριμένα με το ισχυρό Query DSL της Elasticsearch, είναι δυνατόν να πραγματοποιηθούν συνδυαστικές αναζητήσεις full-text (match, multi_match) και δομημένες (term, range) σε δευτερόλεπτα, π.χ. «εντοπισμός όλων των πατεντών με keyword "nanoparticle" δημοσιευμένων μετά το 2018». Ακόμα, για αναφορές και στατιστικά, μπορεί να χρησιμοποιηθεί το aggregation framework (π.χ. $match, $group, $sortByCount) για να υπολογισθεί αριθμός πατεντών ανά έτος, κατά τομέα, ή ανά εφευρέτη. Με αυτόν τον τρόπο, η υλοποίηση αποκτά τόσο ευέλικτη διαχείριση ημι-δομημένων δεδομένων όσο και υψηλής απόδοσης, full-text αναζητήσεις, καθιστώντας δυνατή την εξερεύνηση των υποσυνόλων πατεντών σε πραγματικό χρόνο και σε μεγάλη κλίμακα.

Τέλος, μια χρήσιμη προσθήκη θα ήταν η δημιουργία ενός web-based dashboard με την χρήση του Streamlit που θα επιτρέπει ερευνητές να εξερευνούν διαδραστικά τα υποσύνολα, να φιλτράρουν ανά ημερομηνία, κωδικό ή εφευρέτη, και να εξάγουν γραφήματα on-the-fly. Η διασύνδεση μιας τέτοιας λύσης με την υλοποίηση της παρούσας εργασίας μπορεί να πραγματοποιηθεί αφού υπάρχει υποστήριξη για τέτοιες διεπαφές στην πλατφόρμα Hugging Face όπως έχει αναφερθεί, με τα Spaces

0

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] OECD, *Patents as indicators for innovation*. OECD Publishing,2009.

[2] N. T. Gallini, “The Economics of Patents: Lessons from Recent U.S. Patent Reform,” *Journal of Economic Perspectives*, vol. 16, no. 2, pp. 131–154, 2002.

[3] World Intellectual Property Organization, *WIPO Patent Statistics Manual*, Geneva, 2009.

[4] B. H. Hall, A. B. Jaffe, and M. Trajtenberg, “Market value and patent citations,” RAND Journal of Economics, vol. 36, no. 1, pp. 16–38, 2005.

[5] H. L. Kim, R. Bozeman, “Technology Commercialization Strategy: Analysis of Patent Data,” Technological Forecasting and Social Change, vol. 78, no. 1, pp. 83–95, 2011.

[6] Hugging Face, “Hugging Face Hub,” [Online]. Available: https://huggingface.co/docs/hub

[7] Hugging Face Hub, “Hugging Face Repositories,” [Online]. Available: https://huggingface.co/docs/hub/repositories

[8] Hugging Face Hub, “Hugging Face Datasets Overview,” [Online]. Available: https://huggingface.co/docs/hub/datasets-overview

[9] Hugging Face Hub, “Hugging Face Models,” [Online]. Available: https://huggingface.co/docs/hub/models

[10] Hugging Face Hub, “Hugging Face Spaces,” [Online]. Available: https://huggingface.co/docs/hub/spaces

[11] Open Source Initiative, “The MIT License,” *opensource.org*, 2022. [Online]. Available: https://opensource.org/licenses/MIT

[12] Hugging Face Hub, “Hugging Face Storage limits,” [Online]. Available: https://huggingface.co/docs/hub/storage-limits

[13] Hugging Face Hub, “Hugging Face Dataset Cards,” [Online]. Available: https://huggingface.co/docs/hub/datasets-cards

[14] PyPi, “datasets,” [Online]. Available: https://pypi.org/project/datasets/

[15] Python Data Analysis Library, “User Guide,” [Online]. Available: https://pandas.pydata.org/docs/user_guide/index.html

[16] Hugging Face Hub, “Git vs HTTP paradigm,” [Online]. Available: https://huggingface.co/docs/huggingface_hub/v0.30.2/en/concepts/git_vs_http#git-vs-http-paradigm

[17] Hugging Face Hub, “Cache management,” [Online]. Available: https://huggingface.co/docs/datasets/cache

[18] Hugging Face Hub, “Stream,” [Online]. Available: https://huggingface.co/docs/datasets/stream

[19] Google, "Welcome To Colaboratory," *Google Research*, [Online]. Available: https://colab.research.google.com/

[20] Jupyter, " *Project Jupyter Documentation*,", [Online]. Available: https://docs.jupyter.org/en/latest/

[21] Google Drive, "Google Drive: Share Files Online with Secure Cloud Storage," [Online]. Available: https://workspace.google.com/products/drive/

[22] PyPi, "Pip" [Online]. Available: https://pypi.org/project/pip/

[23] Matplotlib, "Matplotlib: Visualization with Python" [Online]. Available: https://matplotlib.org/

[24] Seaborn, "seaborn: statistical data visualization" [Online]. Available: https://seaborn.pydata.org/

[25] Plotly, "Plotly Express in Python" [Online]. Available: https://plotly.com/python/plotly-express/

[26] Scikit-learn, "CountVectorizer" [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html