Επεξεργασια φυσικης γλωσσας

Εισαγωγει:

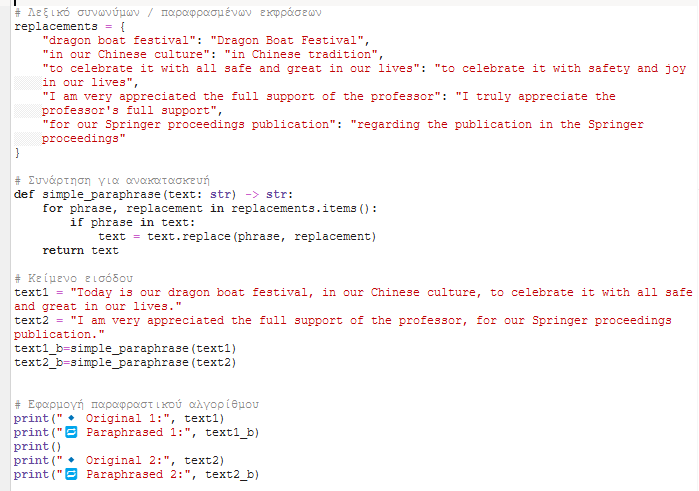
Είναι η διαδικασία όπου παίρνουμε ένα κείμενο και το ξαναγράφουμε (παραφράζουμε) χωρίς να αλλάζει το νόημα του. Αυτό μπορεί να γίνει για λόγους κατανόησης, βελτίωσης του ύφους, αποφυγής λογοκλοπής, ή ακόμα και για μετάφραση.

Για να γινει αυτή η διαδικασεια σωστα πρεπει το NLP να χρησιμοποιη :

* **Ενσωματώσεις λέξεων** (word embeddings) όπως Word2Vec, FastText
* **Μοντέλα νευρωνικών δικτύων** όπως το BERT ή Sentence-BERT
* **Μετρικές ομοιότητας** όπως το cosine similarity και η Levenshtein απόσταση
* **Προεπεξεργασία κειμένου** (lemmatization, stopwords removal κλπ)

Μεθοδολογια:

Α)εδώ χρησιμοποιησα ένα αυτό ματο που εχει καποιες φρασης που όταν της διει θα της αλλαξει από το κειμενο με της καταλειλες άλλες φρασει του αυτοματου .Το αυτό ματο είναι 1 προς 1 εινα hardcoded για της συγκεκριμενες προτασης που επλεξα.



Β)εδώ χρησιμοποιουμε ετιμες βιβλιοθηκες που κανουν ανακατασκεβει κειμενον

1-Spacy Paraphrase:simplify\_text(text) κάνει **απλοποίηση κειμένου** με έναν βασικό, αλλά έξυπνο τρόπο, βασισμένο στη μορφολογική ανάλυση του κειμένου μέσω της βιβλιοθήκης spaCy (nlp).

* **Αν το text είναι λίστα λέξεων** (αντί για string), το ενώνει σε ένα κείμενο.
* Ανάλυση του κειμένου με το spaCy (nlp)

 Παίρνει κάθε λέξη (token) .Αν είναι **ρήμα ή ουσιαστικό**, κρατά το **λήμμα** (π.χ. "running" → "run"). Αλλιώς κρατά τη λέξη όπως είναι

 **Αφαιρεί επαναλήψεις, κρατώντας τη σειρά (με set και sorted(set(...), key=...))**

 **Ενώνει τις λέξεις ξανά σε πρόταση, με κεφαλαίο πρώτο γράμμα και τελεία στο τέλος.**

Επιστρέφει το νέο, απλοποιημένο κείμενο.

2-Τ5 Paraphrase:

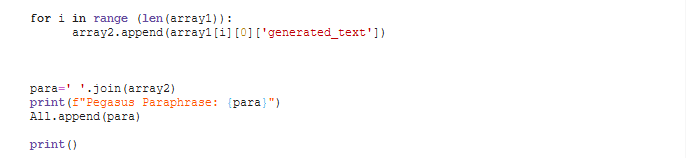
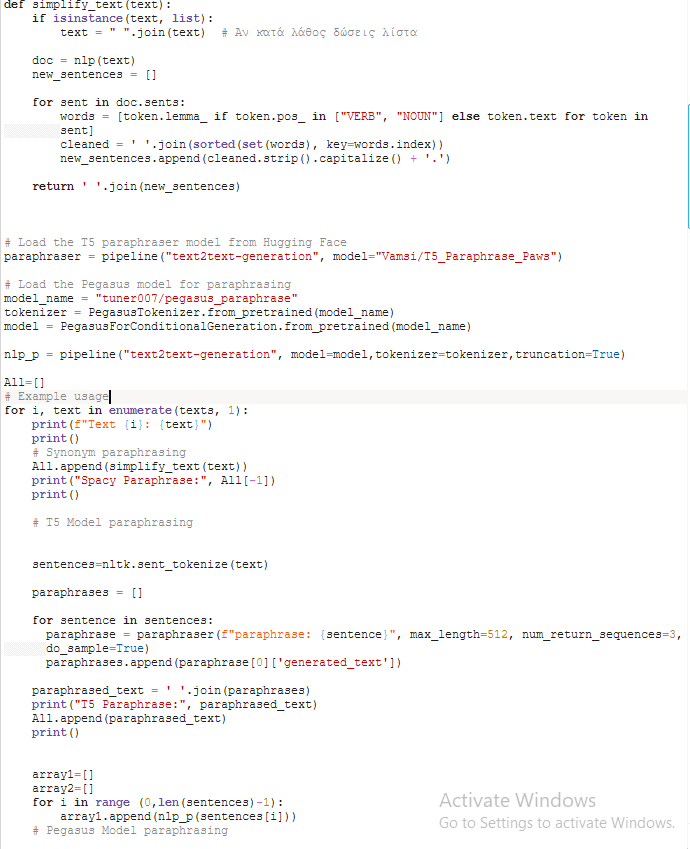
 Χρησιμοποιεί το pipeline του Hugging Face για την εργασία **text2text-generation** (δηλαδή: δώσε κείμενο, πάρε επαναδιατυπωμένο κείμενο).

 Φορτώνει το προκαπροποιημένο μοντέλο "Vamsi/T5\_Paraphrase\_Paws", το οποίο είναι ειδικά εκπαιδευμένο για **παραφράσεις**.

3-Pegasus Paraphrase:

* **Καταλαβαίνει το νόημα** της πρότασης χρησιμοποιώντας attention μηχανισμούς.
* **Αναδομεί το περιεχόμενο** με **διαφορετικές λέξεις και σύνταξη**, χωρίς να αλλάζει το **νόημα**.
* **Παράγει μία ή περισσότερες νέες εκδοχές** που διατυπώνουν την ίδια πληροφορία αλλιώς.

α



C)

**1. Cosine Similarity (Συνάφεια Συνημιτόνου) :**Μετράει τη "γωνιακή" ομοιότητα μεταξύ δύο διανυσμάτων (π.χ. προτάσεων) σε έναν πολυδιάστατο χώρο.

**Πώς λειτουργεί:**

* Κάθε πρόταση μετατρέπεται σε διάνυσμα (π.χ. μέσω Word2Vec, BERT, κ.λπ.).
* Η cosine similarity μετρά το συνημίτονο της γωνίας μεταξύ αυτών των δύο διανυσμάτων.
* Τιμές κοντά στο **1** σημαίνουν **υψηλή ομοιότητα**, στο **0** καθόλου.

**Ιδανική για:** σημασιολογική σύγκριση μεταξύ embeddings.

**2. Levenshtein Distance (Απόσταση Levenshtein):**  
Υπολογίζει πόσες αλλαγές (εισαγωγή, διαγραφή, αντικατάσταση χαρακτήρων) χρειάζονται για να μετατραπεί μια πρόταση σε άλλη.

**Πώς λειτουργεί:**

* Μετρά "επιφανειακή" (συντακτική) ομοιότητα.
* Κανονικοποιείται ώστε να δίνει σκορ ομοιότητας (π.χ. 1 - (distance / max length)).

📌 **Ιδανική για:** απλές συντακτικές αλλαγές – π.χ. τυπογραφικά λάθη ή αλλαγές λέξεων.

### ****3. BERT Similarity:****

Χρησιμοποιεί το μοντέλο BERT για να εξάγει σημασιολογικά διανύσματα (embeddings) από προτάσεις.

**Πώς λειτουργεί:**

* Το BERT αναλύει κάθε λέξη λαμβάνοντας υπόψη το **πλαίσιο (context)** γύρω της.
* Βγάζει ένα ενιαίο διάνυσμα για κάθε πρόταση (μέσος όρος των hidden states).
* Υπολογίζεται η cosine similarity μεταξύ αυτών των διανυσμάτων.

📌 **Ιδανικό για:** σημασιολογική κατανόηση – π.χ. "he bought a car" ≈ "he purchased a vehicle"

### ****4. Word2Vec Similarity:****

Αντιστοιχεί κάθε λέξη σε ένα σταθερό διάνυσμα (χωρίς πλαίσιο), βασισμένο στην ιδέα ότι λέξεις που εμφανίζονται σε παρόμοια συμφραζόμενα έχουν παρόμοια σημασία.

**Πώς λειτουργεί:**

* Κάθε λέξη έχει προκαθορισμένο embedding.
* Η πρόταση εκπροσωπείται από το **μέσο όρο** των διανυσμάτων των λέξεών της.
* Υπολογίζεται cosine similarity μεταξύ των μέσων διανυσμάτων δύο προτάσεων.

📌 **Πλεονέκτημα:** Ελαφρύ και γρήγορο  
📌 **Μειονέκτημα:** Δεν καταλαβαίνει το **πλαίσιο** ή τη **συντακτική δομή**.

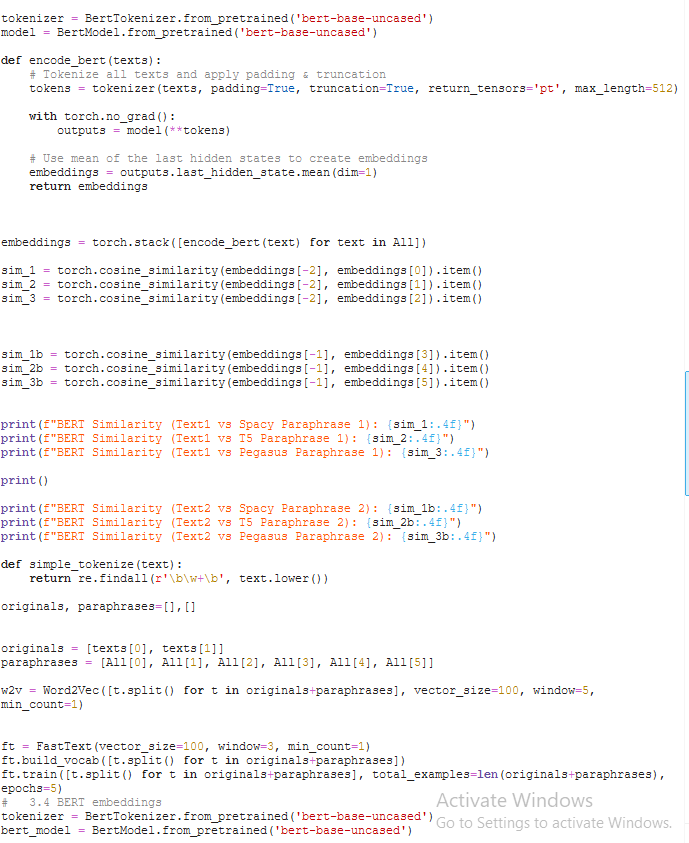
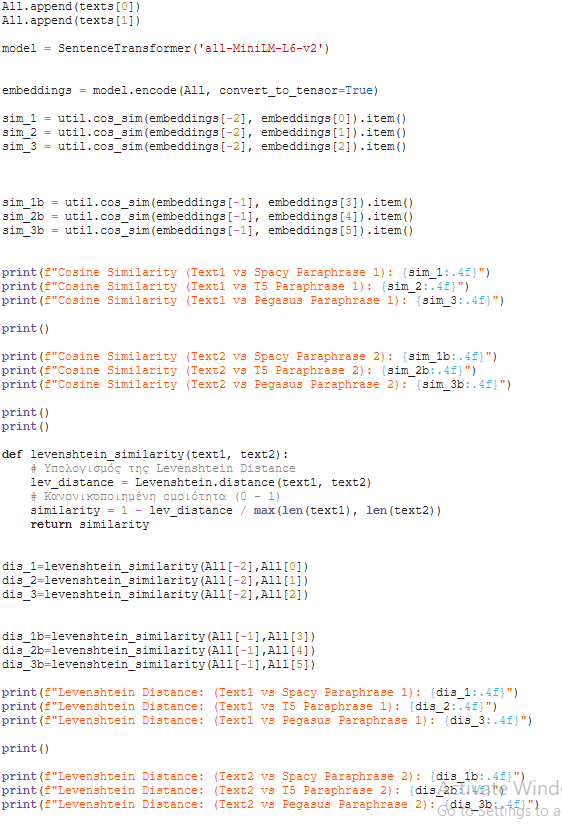
### ****5. FastText Similarity:****

Παρόμοιο με Word2Vec αλλά λαμβάνει υπόψη **υπολέξεις (subwords)**, οπότε αναγνωρίζει και άγνωστες λέξεις ή παραλλαγές (π.χ. "playing", "player").

**Πώς λειτουργεί:**

* Κάθε λέξη αναλύεται σε μικρότερα κομμάτια (n-grams).
* Χρησιμοποιείται ο μέσος όρος των διανυσμάτων των λέξεων ή υπολεξικών μονάδων.
* Υπολογίζεται cosine similarity.

📌 **Πλεονέκτημα:** Ανθεκτικό σε σπάνιες/νέες λέξεις  
📌 **Ιδανικό για:** γλώσσες με πλούσια μορφολογία.



a



2)εδώ κανουμε preporoses τα ανακατασκεβασμενα κειμενα και ξανατρεχουμε της υπολογιστηκες τεχνηκες ώστε να δουμε αμα εχουμε καλητερα αποτελεσματα πριν η μετα το preproses

### ****Συνάρτηση**** preprocess(text)

* Κάνει το κείμενο **πεζό** (lowercase)
* Χρησιμοποιεί το spaCy (nlp) για να:
  + **Λεματοποιήσει** τις λέξεις (π.χ. "running" → "run")
  + **Αφαιρέσει stopwords** (όπως "the", "and", "is")
  + **Αφαιρέσει σημεία στίξης**
* Επιστρέφει λίστα από λέξεις (ή tokens).

Επιτα κανουμε της ιδιες διαδικασειες με πριν και εχουμε

def preprocess(text):

doc = nlp(text.lower())

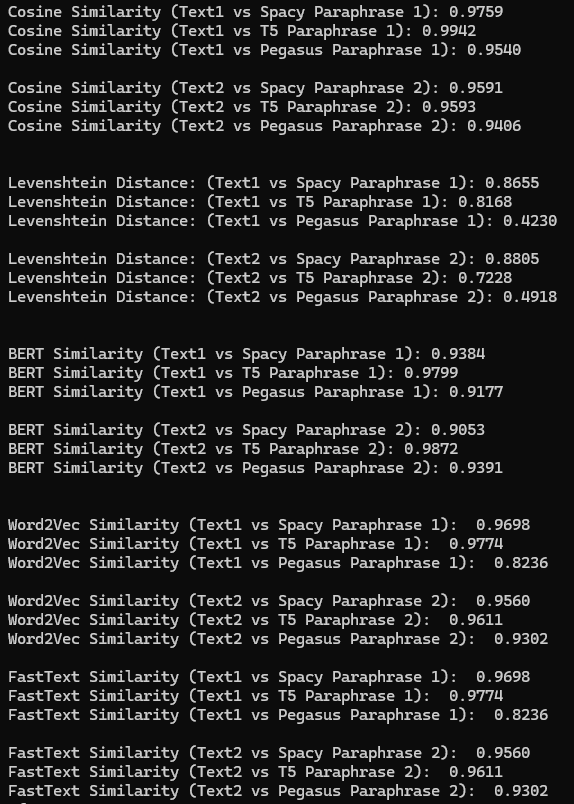
tokens = [t.lemma\_ for t in doc if not t.is\_stop and not t.is\_punct]

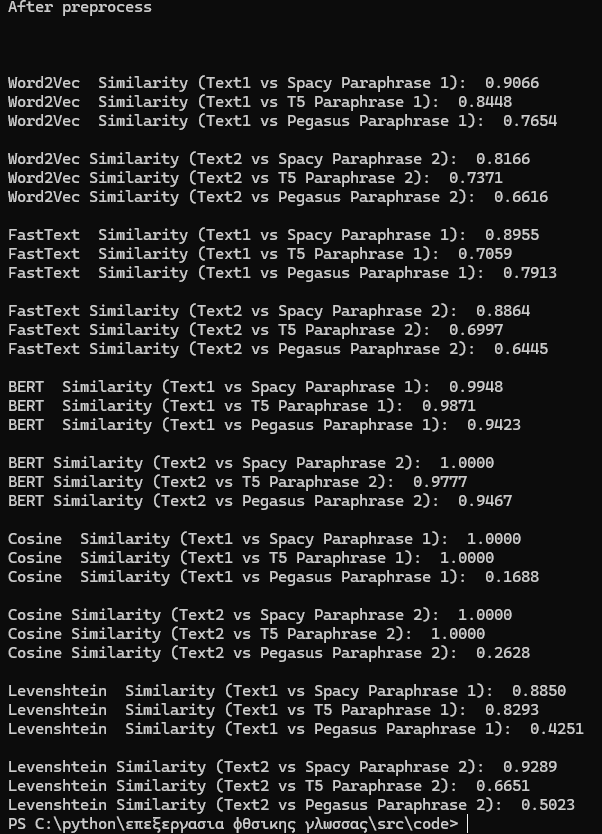
if not tokens:

tokens = [t.text for t in doc if not t.is\_punct]

return tokens

παρομια γιανεται και για το Α)





## **BERT Similarity – Πιο Σταθερή και Υψηλή**

* **Πριν & μετά** την προεπεξεργασία, το **BERT** διατηρεί **πολύ υψηλές τιμές** (πάνω από 0.94), γιατί καταλαβαίνει **συμφραζόμενα και νόημα**.
* Μετά το preprocessing, οι τιμές του BERT αυξάνονται ελαφρώς → υποδηλώνει **καθαρότερο και πιο συνεκτικό κείμενο**.

🔎 **Συμπέρασμα**: Το BERT είναι **ανθεκτικό στην προεπεξεργασία** και αποδίδει εξαιρετικά με σημασιολογική προσέγγιση.

## **2. Cosine Similarity (TF-IDF based)**

* **Πριν το preprocessing**: Πολύ καλές τιμές (~0.95–0.99).
* **Μετά το preprocessing**: Για T5 & Spacy παραφράσεις = **1.0000**, δηλαδή τα κείμενα είναι σχεδόν **πανομοιότυπα** στο λεξιλόγιο.
* Όμως για Pegasus: **τεράστια πτώση (0.16–0.26)** → ένδειξη ότι το Pegasus αλλάζει αρκετά τη μορφή των λέξεων.

🔎 **Συμπέρασμα**: Το preprocessing ενισχύει τη λεξιλογική ομοιότητα (ειδικά με απλά paraphrasers όπως Spacy ή T5), αλλά μπορεί να **υπονομεύσει** τη μέτρηση σε πιο δημιουργικά μοντέλα (π.χ. Pegasus).

## ✅ **3. Levenshtein Distance – Αλλαγή σε χαρακτήρες**

* Προσμετρά **την επεξεργασία χαρακτήρων** (πόσες αλλαγές χρειάζονται).
* Οι τιμές **βελτιώνονται ελαφρώς** με την προεπεξεργασία (ιδιαίτερα για Spacy), αλλά **πέφτουν για Pegasus**, λόγω μεγαλύτερων αλλαγών.

🔎 **Συμπέρασμα**: Το Levenshtein δείχνει ότι το preprocessing οδηγεί σε πιο "παρόμοιο" κείμενο σε επίπεδο χαρακτήρων, **μόνο όταν οι παραφράσεις είναι κοντινές** στο πρωτότυπο.

## ✅ **4. Word2Vec & FastText**

* **Πριν**: Υψηλές τιμές (~0.95–0.97 για Spacy/T5).
* **Μετά**: Σημαντική **πτώση** (έως και ~0.66 για Pegasus).
* Αυτό συμβαίνει γιατί Word2Vec και FastText χρειάζονται **πλούσιο και αυθεντικό κείμενο** — η προεπεξεργασία μειώνει τη διαφοροποίηση και **χάνει συμφραζόμενα**.

🔎 **Συμπέρασμα**: Τα embeddings χάνουν "πλούτο" όταν αφαιρούμε stopwords και κάνουμε λεματοποίηση, άρα **προσεκτική χρήση** του preprocessing όταν χρησιμοποιούμε τέτοιες τεχνικές.

Συζητηση:

### Πόσο καλά αποτύπωσαν οι ενσωματώσεις λέξεων το νόημα;

Οι ενσωματώσεις λέξεων (Word2Vec, FastText, BERT) είχαν **διαφορετικά επίπεδα επιτυχίας**:

* **BERT** παρουσίασε **την καλύτερη απόδοση**, καθώς βασίζεται σε συμφραζόμενα και κατανοεί το νόημα των προτάσεων ακόμα και όταν η διατύπωση αλλάζει σημαντικά. Οι τιμές ομοιότητας παρέμειναν υψηλές ακόμη και μετά την προεπεξεργασία, υποδεικνύοντας **ισχυρή σημασιολογική αναπαράσταση**.
* **Word2Vec και FastText** προσφέρουν **καλές αλλά όχι τέλειες** σημασιολογικές αναπαραστάσεις. Επηρεάζονται από την απώλεια stopwords και τη λεμματοποίηση, γεγονός που μειώνει την ακρίβεια των μετρήσεων. Παρά τη γενική τους αποτελεσματικότητα, δεν αποδίδουν τόσο καλά όσο τα σύγχρονα μοντέλα βασισμένα σε μετασχηματιστές (transformers).

### Ποιες ήταν οι μεγαλύτερες προκλήσεις στην ανακατασκευή;

1. **Διατήρηση νοήματος**: Η μεγαλύτερη πρόκληση ήταν η παραγωγή παραφρασμένων προτάσεων που **διατηρούν το αρχικό νόημα** χωρίς να είναι λέξη-προς-λέξη όμοιες.
2. **Διαχείριση διαφορετικών στυλ παραφράσεων**: Κάποια μοντέλα (όπως Pegasus) επιλέγουν πιο δημιουργική ή αφαιρετική διατύπωση, κάνοντάς τα πιο δύσκολα στη σύγκριση με λεξιλογικές μεθόδους (TF-IDF, Levenshtein).
3. **Προεπεξεργασία και σημασιολογία**: Ενώ η προεπεξεργασία βοηθά στην αφαίρεση θορύβου, **μπορεί να αφαιρέσει και χρήσιμο πλαίσιο**, δυσκολεύοντας τα embedding μοντέλα να κατανοήσουν πλήρως το νόημα.

### Πώς μπορεί να αυτοματοποιηθεί αυτή η διαδικασία χρησιμοποιώντας μοντέλα NLP;

Η διαδικασία μπορεί να αυτοματοποιηθεί με τα εξής βήματα:

1. **Αυτόματη Παραφράση**: Χρήση μοντέλων όπως **T5**, **Pegasus**, ή **GPT** για τη δημιουργία παραφράσεων.
2. **Αξιολόγηση Ομοιότητας**: Ενσωμάτωση μεθόδων σύγκρισης όπως **BERT similarity**, **Cosine TF-IDF**, και **Levenshtein**, ώστε να υπολογίζονται αυτόματα οι αποστάσεις μεταξύ πρωτοτύπων και παραφράσεων.
3. **Φιλτράρισμα/Επιλογή**: Εφαρμογή φίλτρων (thresholds) για να επιλέγονται οι πιο ποιοτικές παραφράσεις ή να απορρίπτονται όσες διαφέρουν πολύ.
4. **Pipeline NLP**: Η σύνδεση αυτών των εργαλείων σε μία pipeline επιτρέπει **ενιαία και επαναλαμβανόμενη ανακατασκευή** σε μεγάλα κείμενα ή datasets.

### Υπήρξαν διαφορές στην ποιότητα ανακατασκευής μεταξύ τεχνικών, μεθόδων, βιβλιοθηκών;

Ναι, και ήταν **πολύ εμφανείς**:

* **Pegasus** προσέφερε **πιο φυσική αλλά πιο απομακρυσμένη** παραφράση, άρα χαμηλότερη ομοιότητα σε επιφανειακές μετρήσεις (Levenshtein, TF-IDF), αλλά καλό νόημα (BERT).
* **T5** έδωσε **ισορροπημένες παραφράσεις** — ούτε υπερβολικά κοντές, ούτε υπερβολικά αφαιρετικές. Πέτυχε **υψηλή συνολική απόδοση** σε όλες τις μετρήσεις.
* **SpaCy (λεμματοποίηση + reorder)** δεν παράγει παραφράσεις με νοηματική ποικιλία, αλλά οδηγεί σε **υψηλή λεξιλογική ταύτιση**. Κατάλληλο για «συντηρητικές» επαναδιατυπώσεις.

### Συμπεράσματα

* Οι **σύγχρονες τεχνικές NLP**, ειδικά τα μοντέλα με βάση τους **transformers**, έχουν **μεγάλη δυνατότητα σημασιολογικής ανακατασκευής**.
* Η **προεπεξεργασία** μπορεί να **βλάψει** τη σύγκριση σε απλές μετρικές αλλά **να ενεισχηση** τα embeddings.
* Η **επιλογή του κατάλληλου μοντέλου** εξαρτάται από το τι επιδιώκουμε (ακρίβεια vs ποικιλία).
* Ο συνδυασμός **πολλαπλών μετρικών ομοιότητας** είναι απαραίτητος για πιο αξιόπιστη αξιολόγηση παραφράσεων.

Βιβλιογραφια: chatGPT,thales