# ΑΝΑΛΥΣΗ ΕΙΚΟΝΑΣ

**Εργασία 2023/24**


**Ομάδα:**

**Κυριάκος Κίχης, Π20013**

**Χρίστος Ξύδης, Π20008**

## i. Αναλυτική περιγραφή της υπολογιστικής διαδικασίας

Ο αλγόριθμος που προτείνεται και περιγράφεται στο προηγούμενο άρθρο είναι γνωστός ως "Log-Based Hypergraph of Ranking Reference" και αποτελείται από πέντε βασικά στάδια τα οποία παρουσιάστηκαν στη συγκεκριμένη μελέτη. Για την εκτέλεση του αλγορίθμου, πρέπει πρώτα να προκύψει το σύνολο των αντικειμενικών χαρακτηριστικών για τη διανυσματική αναπαράσταση κάθε πολυμεσικού αντικειμένου. Στο άρθρο αναφέρεται μια συνάρτηση εξαγωγής χαρακτηριστικών ε. Στην πραγματικότητα, αυτή η διαδικασία υλοποιείται μέσω ενός προ-εκπαιδευμένου μοντέλου του torchvision. Έχουμε τροποποιήσει το επίπεδο εξόδου αντικαθιστώντας το με ένα επίπεδο χωρίς λειτουργία, επιτρέποντας μας να εξάγουμε τα χαρακτηριστικά απευθείας από το τελευταίο κρυφό επίπεδο. Πριν αναλυθούν τα στάδια του αλγορίθμου, πρέπει να καθοριστούν ορισμένες έννοιες. Αρχικά, ο αλγόριθμος χρησιμοποιεί ένα σύνολο C = {o1, o2, ..., on}, όπου το κάθε on είναι μια εικόνα στο dataset και |C| = n, όπου n είναι ο αριθμός των εικόνων του dataset. Ο στόχος του αλγορίθμου είναι για ένα στοχείο στόχο oq να βρεθούν τα k πλησιέστερα στοιχεία. Αυτό επιτυγχάνεται μέσω μιας λίστας Tq = {tq(1), tq(2), ..., tq(n)}, όπου tq(i), i ∈ [N], αναπαριστά τη θέση του στοιχείου oi στην ταξινομημένη λίστα του oq. Για την ταξινόμηση αυτής της λίστας χρησιμοποιείται η συνάρτηση ομοιότητας ρ. Έτσι, αν σε μια λίστα Tq, το oi προηγείται του oj, αυτό συνεπάγεται ότι tq(i) < tq(j) και, συνεπώς, ρ(q, i) >= ρ(q, j). Στην υλοποίησή μας, η αρχική συνάρτηση ομοιότητας ήταν το αντίστροφο της Ευκλείδειας απόστασης, όπως προτάθηκε και στην τάξη. Τέλος, καθορίζονται τα γειτονικά σύνολα N(q, k), τα οποία περιλαμβάνουν τα k πλησιέστερα στοιχεία ως προς το στοιχείο στόχο oq. Αξίζει να σημειωθεί ότι επειδή η δημιουργία λιστών για μεγάλα σύνολα από εικόνες είναι μια αργή διαδικασία με υψηλό κόστος, το άρθρο προτείνει να κρατηθούν μόνο τα L καλύτερα στοιχεία σε ένα νέο υποσύνολο CL ⊂ C. Για ευκολία, από εδώ και στο

εξής θα χρησιμοποιούμε τον όρο "εικόνα" αντί για "πολυμεσικό αντικείμενο", δεδομένου ότι η μελέτη και το μάθημα αναφέρονται σε εικόνες.

**Βήματα αλγορίθμου:**

**1. Κανονικοποίηση σειράς κατάταξης:** Πραγματοποιείται μια διαδικασία κανονικοποίησης για κάθε λίστα Ti όπου υπολογίζεται ένα νέο similarity metric ρn (i,j) = 2L - ( τi( j ) + τi( i ) ) με σκοπό να βελτιώσει τη συμμετρία των N(q,k) σχέσεων γειτονιάς. Στη συνέχεια η κάθε λίστα ταξινομείται.

**2. Κατασκευή υπεργραφήματος:** Το υπεργράφημα είναι μια γενίκευση ενός κανονικού γραφήματος όπου οι ακμές αποτελούνται από μη κενά υποσύνολα του συνόλου V των κορυφών, επιτρέποντας τη σύνδεση οποιωνδήποτε ακμών μεταξύ τους. Για ένα γράφημα G = (V, E, w), το V είναι το σύνολο των κορυφών, το E το σύνολο των υπερακμών, όπου κάθε Ue ∈ E = V. Μια υπερακμή ei είναι ένα σύνολο κορυφών ei = {u1, u2, ..., un}, ορισμένο με βάση το k γειτονικό σύνολο και τους αντίστοιχους γείτονες. Στη δική μας υλοποίηση, κάθε υπερακμή είναι "κεντραρισμένη" σε μια εικόνα/κόμβο, με τον πρώτο κόμβο στο σύνολο κορυφών της ei να είναι ο vi. Η συμμετοχή μιας εικόνας/κόμβου σε μια υπερακμή μπορεί να οριστεί δυαδικά, αλλά στο άρθρο και στην υλοποίησή μας χρησιμοποιούμε πιθανοτική αναπαράσταση. Αν έστω ox ∈ N(i, k) είναι ένας γείτονας του oi και oj ∈ N(x, k) είναι ένας γείτονας του ox, το μέτρο συμμετοχής r(ei, uj) της κορυφής uj στην υπερακμή ei υπολογίζεται ως εξής: r(ei, uj) = Σox ∈ N(i, k) ^ oj ∈ N(x, k) wp(i, x) x wp(x, j), όπου wp(i, x) είναι μια συνάρτηση που αναθέτει ένα βάρος στο ox βάσει της θέσης του στην ταξινομημένη λίστα ti. Το βάρος που ανατίθεται στην εικόνα ox βάσει της θέσης της στη λίστα ti υπολογίζεται ως εξής: wp(i, x) = 1 - logk ti(x). Αυτή η συνάρτηση δίνει ένα μέγιστο βάρος ίσο με 1 στην πρώτη θέση της λίστας (που είναι η ίδια η εικόνα στόχος) και μειώνει ταχέως τα βάρη για τις επόμενες ταξινομημένες θέσεις. Στόχος είναι να ανατεθούν υψηλά βάρη στις πρώτες θέσεις, όπου η αποτελεσματικότητα των ταξινομημένων λιστών είναι καλύτερη. Επιπλέον, στη δική μας υλοποίηση, το βάρος υπολογίζεται με τον τύπο wp(ei, vj) = 1 - logk+1 ti(j), ο οποίος αποτελεί διαφοροποίηση από το άρθρο.

Με βάση τις τιμές r(ei, uj), ορίζουμε τον πίνακα εγκυρότητας ως εξής: H(ei, vj) = r(ei, vj), αν vj ∈ ei, ενώ διαφορετικά H(ei, vj) = 0. Επιπλέον, πρέπει να οριστεί ένα βάρος w(ei) για κάθε υπερακμή ei, το οποίο δηλώνει την αυτοπεποίθηση των σχέσεων που έχουν δημιουργηθεί ανάμεσα στις κορυφές των υπερακμών. Όσο μικρότερο το πλήθος και η ποικιλία των κορυφών της υπερακμής, τόσο καλύτερη η ποιότητα των υπερακμών και οι τιμές του h(ei). Το w(ei) υπολογίζεται για κάθε υπερακμή βάσει των κορυφών που ανήκουν σε αυτήν: w(ei) = Σj ∈ Nh(i, k) H(i, j).

**3. Υπολογισμός ομοιότητας υπερακμών:** Αρχικά, πρέπει να υπολογιστεί ο πίνακας ομοιότητας μεταξύ ζευγαριών υπερακμών S. Για να γίνει αυτό, πρέπει να οριστούν δύο επιπλέον πίνακες βάσει δύο υποθέσεων. Η πρώτη υπόθεση είναι ότι παρόμοιες εικόνες παρουσιάζουν παρόμοιες ταξινομημένες λίστες και επομένως παρόμοιες υπερακμές. Δεδομένου ότι όλη η πληροφορία σχετικά με τη συσχέτιση των υπερακμών έχει κωδικοποιηθεί στον πίνακα εγκυρότητας, μια μέτρηση ομοιότητας για δύο υπερακμές ei και ej μπορεί να οριστεί ως εξής: Sh = HHT. Η δεύτερη υπόθεση υποδηλώνει ότι παρόμοιες εικόνες αναμένεται να αναφέρονται από τις ίδιες υπερακμές, και από αυτήν προκύπτει ο πίνακας Su = HTH. Εφόσον και οι δύο πίνακες ομοιότητας περιέχουν σχετική πληροφορία, το pairwise similarity matrix S μπορεί να υπολογιστεί μέσω του Hadamard product (στοιχειοτικός πολλαπλασιασμός): S = Sh ◦ Su.

**4. Υπολογισμός καρτεσιανού γινομένου μεταξύ των υπερακμών:** Για να εξαχθούν σχέσεις ζευγαριών απευθείας από το σύνολο των στοιχείων που ορίζουν οι υπερακμές, πρέπει να υπολογιστεί το καρτεσιανό γινόμενό τους. Καθώς κάθε υπερακμή συνδέει ένα σύνολο κορυφών, το καρτεσιανό γινόμενο ορίζεται ως εξής: eq 2 = eq x ei = { (ux ,uy ) : ux ∈ eq ^ uy ∈ ei }. Επομένως, για κάθε ζεύγος κορυφών που ανήκουν στο καρτεσιανό γινόμενο των υπερακμών ορίζεται μια σχέση σχετικότητας κατά ζεύγη. Στη συνέχεια, υπολογίζεται εκ νέου η συνάρτηση ρ με βάση το w(eq), η οποία διατυπώνει την αυτοπεποίθηση της υπερακμής eq που ορίζει τη συσχέτιση. Ο βαθμός συμμετοχής των κορυφών ui, uj είναι: ρ(eq,ui,uj) = w(eq) x H(eq,ui) x H(eq,uj). Το similarity measure του καρτεσιανού γινομένου υπολογίζεται μέσω ενός πίνακα C που λαμβάνει υπόψη τις σχέσεις που υπάρχουν σε όλες τις υπερακμές. Η σκέψη πίσω από αυτούς τους

υπολογισμούς βασίζεται στην πιθανότητα συνύπαρξης των κορυφών ui και uj σε διαφορετικές υπερακμές. Κάθε θέση στον πίνακα C υπολογίζεται ως εξής: C(i,j) = Σeq∈E^(ui,uj)∈ eq 2 ρ(ui,uj).

**5. Υπολογισμός ομοιότητας βάσει του κατασκευασμένου υπεργράφου**: Το μέτρο ομοιότητας που υπολογίστηκε με βάση τις υπερακμές και το καρτεσιανό γινόμενο παρέχει πληροφορία για την ποικιλομορφία των δεδομένων. Αυτή η πληροφορία συνδυάζεται μέσω ενός πίνακα συγγένειας W, που πολλαπλασιάζει τους πίνακες C και S: W = COS

Με βάση το βαθμό συγγένειας που υπολογίστηκε στον πίνακα W, εφαρμόζεται μια διαδικασία κατάταξης για τη δημιουργία μιας νέας ταξινομημένης λίστας T. Η διαδικασία είναι η εξής: Έστω T(0) η αρχική λίστα που προέκυψε από τα αντικειμενικά χαρακτηριστικά. Η λίστα T(t+1) υπολογίζεται βάσει του W (t). Μετά από έναν συγκεκριμένο αριθμό επαναλήψεων που καθορίζουμε στο πρόγραμμα, υπολογίζεται η τελική ταξινομημένη λίστα T' για κάθε εικόνα που ανήκει στο σύνολο Γ.

## ii. <u>Προγραμματιστική Υλοποίηση</u>

Ο αλγόριθμος υλοποιήθηκε σε python, σε περιβάλλον Visual Studio Code

● Για την υλοποίηση χρησιμοποιήθηκαν οι βιβλιοθήκες pytorch (+torchvision), matplotlib, numpy και hypernetx*.

● Ως σύνολο δεδομένων χρησιμοποιούμε το σύνολο δεδομένων Caltech101 (συγκεκριμένα ένα υποσύνολο του για να μειωθεί ο χρόνος εκτέλεσης). Το dataset αυτό περιέχεται στα datasets του torchvision και εγκαθίσταται μέσω της βιβλιοθήκης.

● Για την εξαγωγή χαρακτηριστικών χρησιμοποιήσαμε ένα από τα διαθέσιμα προεκπαιδευμένα μοντέλα της βιβλιοθήκης torchvision.

● Επιπλέον χρησιμοποιήσαμε ένα αρχείο από ένα github repository με μια βοηθητική συνάρτηση για την προβολή εικόνων από το dataset (helper.py).

*η hypernetx αρχικά χρησιμοποιήθηκε για την κατασκευή ενός hypergraph object, αλλά τελικά συμπεράναμε ότι δεν χρειάζεται στη υλοποίησή μας.

Βασικό κομμάτι στην υλοποίησή μας είναι ότι δεν θεωρούμε συγκεκριμένες εικόνες ως εικόνες query, αλλά δεδομένου ενός συνόλου εικόνων μεγέθους π.χ. 3000, υπολογίζουμε similarities κλπ. για όλες τις εικόνες μεταξύ τους. Ο τελικός πίνακας δηλαδή, στην προκειμένη περίπτωση, έχει μέγεθος 3000x3000 και η i-οστή γραμμή αντιστοιχεί στα σκορ της i-οστής εικόνας με τις υπόλοιπες. Με τον τρόπο αυτό μπορούμε στο τέλος να χρησιμοποιήσουμε ως εικόνα query οποιαδήποτε από τις 3000 εικόνες θέλουμε και να κάνουμε retrieve τις εικόνες με τα καλύτερα σκορ στην αντίστοιχη γραμμή.

## iii. Εξαγωγή χαρακτηριστικών

Για την εξαγωγή χαρακτηριστικών από τις εικόνες, χρησιμοποιήσαμε ένα προεκπαιδευμένο νευρωνικό δίκτυο τύπου Vision Transformer (vit_l_16 στο torchvision). Το δίκτυο αυτό έχει εκπαιδευτεί στο σύνολο δεδομένων ImageNet (για classification).

Αφού κατεβάσουμε τα τελικά βάρη (1.2 GB) και κατασκευάσουμε το δίκτυο με βάση αυτά, απενεργοποιούμε τον υπολογισμό των gradients για να μην αλλάξουν τα βάρη, γιατί δεν μας ενδιαφέρει να το εκπαιδεύσουμε περαιτέρω. Στη συνέχεια τυπώνουμε το μοντέλο για να μάθουμε περισσότερα για τα ενδιάμεσα επίπεδα του. Για να εξάγουμε τα χαρακτηριστικά, πρέπει να αντικαταστήσουμε το τελευταίο επίπεδο (που αποτυπώνει σε ποια κλάση από τις ~1000 της βάσης ImageNet ανήκει μια εικόνα) με ένα τευτονικό επίπεδο (identity layer) ή ένα άδειο Sequential layer που δεν κάνει τίποτα. Εμείς κάνουμε το δεύτερο και με τον τρόπο αυτό η έξοδος που προκύπτει είναι η έξοδος του τελευταίου κρυφού επιπέδου του δικτύου. Αυτά τα χαρακτηριστικά χρησιμοποιούνται ως αρχική είσοδος του αλγορίθμου.

## iv. Παραδείγματα Εκτέλεσης

Παράδειγμα 1:

Πάνω έχουμε την εικόνα που δώσαμε για εκτέλεση (query image) και κάτω έχουμε αυτές που μας δόθηκε ότι μοιάζουν περισσότερο με αυτήν (retrieved images), ξεκινώντας από την πιο αριστερά. Αυτό ισχυεί για όλα τα παραδείγματα.

Scores, IDs:

```
2.080244065835263 1207
0.6047548240938942 1804
0.03364256203168948 682
0.015259401746890366 1451
0.004595790071199779 1385
```

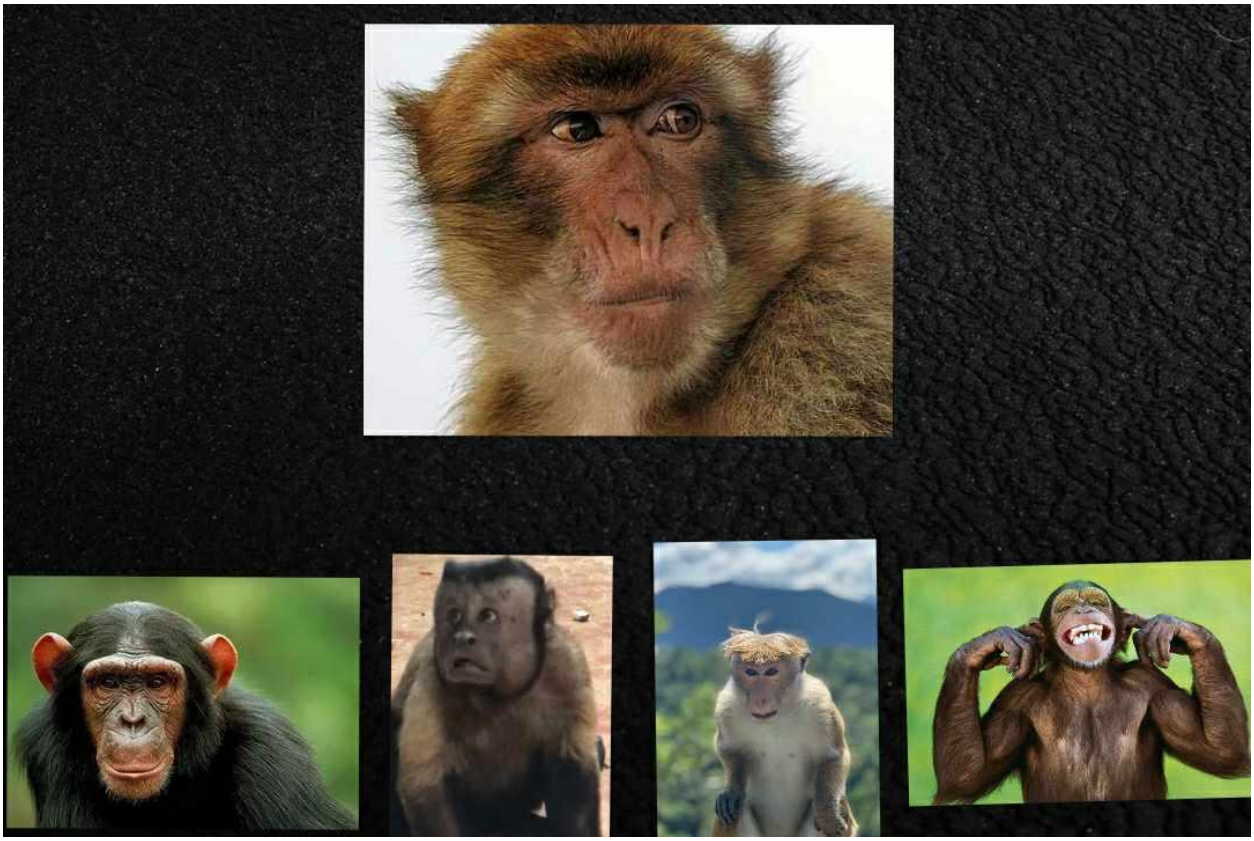
Παράδειγμα 2:

Scores, IDs:

```
1.7075187496394222 652
0.21343984370492775 1036
0.12207521397512293 1210
```

```
0.21343984370492775 143
0.015259401746890366 467
```

Παράδειγμα 3:



Scores, IDs:

```
1.707518749639422 832
0.015259401746890366 437
```

```
0.213439843704927 75 1096
0.03364256203168948 5 639
0.015259401746890366 1842
```

## v. Μέτρηση ακρίβειας

Για την μέτρηση της ακρίβειας του αλγορίθμου, μιας και η διαδικασία πρόκειται για Information Retrieval task, επιλέξαμε να χρησιμοποιήσουμε 2 μετρικές που χρησιμοποιούνται συχνά σε τέτοια συστήματα: Precision και Recall. Το σύνολο δεδομένων που χρησιμοποιήσαμε μαζί με τις εικόνες παρέχει και τα true labels τους (σε ποια κατηγορία ανήκουν επειδή το dataset χρησιμοποιείται κανονικά για classification). Εμείς χρησιμοποιούμε αυτά τα labels για να υπολογίσουμε το precision και το recall.

Για τον υπολογισμό του precision, αν ο αλγόριθμος είχε ως έξοδο n όμοιες εικόνες για μια query εικόνα υπολογίζουμε πόσες από αυτές τις εικόνες έχουν ίδιο label με την εικόνα query.

Precision = (retrieved images with correct label) / (total retrieved images)

Για τον υπολογισμό του recall χρησιμοποιούμε τα labels που μας παρέχει το dataset με παρόμοιο τρόπο. Συγκεκριμένα, δεδομένου του label της query εικόνας, υπολογίζουμε το ποσοστό των εικόνων με ίδιο label που έγιναν retrieved, από όλες τις εικόνες με αυτό το label.

Recall = (retrieved images with correct label) / (total images with correct label)

Αξίζει να σημειωθεί ότι οι τιμές του recall που υπολογίζονται είναι αρκετά χαμηλές. Αυτό συμβαίνει επειδή ως σύνολο δεδομένων χρησιμοποιούμε αριθμό εικόνων της τάξης του 1000-3000 εικόνες, αλλά ως μέγεθος υπερακμής στα παραδείγματα χρησιμοποιούμε μικρούς αριθμούς (3-5). Το μέγεθος υπερακμής δεν περιορίζει αυστηρά τον αριθμό των retrieved εικόνων (μπορεί να είναι περισσότερες από 3-5) αλλά παρατηρήσαμε ότι όσο 14 αυξάνεται, γενικά αυξάνονται και οι retrieved εικόνες. Τα recall που υπολογίζονται είναι χαμηλά επειδή μπορεί να έχουν γίνει retrieved π.χ. 5 εικόνες με κάποιο label αλλά στο σύνολο των δεδομένων που χρησιμοποιούμε να υπάρχουν 100+ εικόνες με το ίδιο label. Επίσης όπως εξηγήθηκε και πιο πάνω, το νευρωνικό δίκτυο που χρησιμοποιούμε είναι ακριβές σε σημείο που ξεχωρίζει λεπτομέρειες. Αυτό έχει

ως αποτέλεσμα συχνά να διαφοροποιεί εικόνες με ίδιο label αλλά μικρές διαφορές μεταξύ τους.