

תיק פרויקט

# Automatic Document Sorter

סיווג מסמכים אוטומטי

מגיש : עמרי גיגי ת.ז. 211881396

מגיש : לב ת.ז. 209780881

מנחה : שחר אוחנה

תואר : הנדסאי תוכנה

תאריך :

## תוכן עניינים

6	הסבר הפרויקט
6	בעיות במערכת
6	פתרון הבעיות של המערכת
6	תיאור הבעיה האלגוריתמית
6	מטרות הפרויקט
7	רקע תיאורטי בתחום הפרויקט
7	- nlp
7	רשת עצבית מלאכותית-
8	תהליכים עיקריים בפרויקט
9	תיאור המערכת
10	צד שרת
10	צד לקוח
11	מסד הנתונים
11	דרישות עמדת פיתוח
11	דרישות מינימליות לעמדת המשתמש
11	פרוטוקול תקשורת
11	ביבליוגרפיה
12	לוחות זמנים
13	תקציר / מבוא
13	רקע הפרויקט
13	תהליך המחקר
13	סקירה ספרות
14	אתגרים מרכזיים
14	הבעיה איתה מתמודד התלמיד
14	הנגשת המידע למכונה
14	ניקוי המידע
14	בניית המוכנה
16	הסיבה לבחירת הנושא
16	מוטיבציה לעבודה
17	הצרכים שהפרויקט עונה אליהם
17	הפתרון לבעיה יצירת המידע
19	הפתרון לבעיה יצירת המודל
21	מטרות ויעדים ומדדים להצלחה
21	מטרות
21	יעדים
22	אתגרים
24	מדדי הצלחה למערכת
24	רקע תיאורטי

24	תיאור מצב קיים
24	ניתוח חלופי למערכת
25	תיאור החלופה הנבחרת
26	אפיון המערכת שהוגדרה
26	ניתוח דרישות המערכת
27	מודל מערכת- תרשים זרימה לוגית של המערכת :
28	אפיון פונקציונלי
28	פירוק pdf למילים
28	ניקוי ותרגום המידע
28	מיון מילים לפי שכיחותם
28	מערכת לניהול תהליכים
28	יכולת חיזוי המסמך
29	הוצאת הנותנים המסמך
29	סידור מסמכים בתיקיות
29	לולאת אוטומציה
29	עידכון בסיס הנתונים
29	התחברות והתנתקות
29	הגדרות
30	ביצועים עיקריים
30	אילוצים
30	סביבת פיתוח :
30	כלי עזר :
31	תיכון ארכיטקטורת המערכת
31	תרשים כללי ארכיטקטורת המערכת
32	תיאור הרכיבים בפתרון
32	תיאור התקשורת
33	ניתוח ותרשים Use / UML cases של המערכת המוצעת
33	תרשים Use Case
34	תרשים UML של הממשק הגרפי
35	תרשים UML של AI
35	תרשים UML של Servicen (שרת לוקלי)
36	תיאור המחלקות המוצעות
36	צד לקוח-ממשק גרפי
37	צד שרת
38	מבנה נתונים :
38	מילון :
38	שימוש במילון :
38	מערך בולאני :
39	מדריך למשתמש-תיאור GUI (מסכים וכפתורים)
39	LogIn

39	החלון המרכזי
41	Menu
41	Settings
42	חלק ראשון - Primary
44	חלק שני - Networking
45	Manual Sorting
45	חלק ראשון :
46	חלק שני :
46	חלק שלישי :
46	חלק רביעי :
47	Notifications
48	ניתוח קוד התוכנית
48	ספר 1 : יצירת וקטור מילים
48	import
48	הוצאת מילים מתמונה
49	פעולת הוצאת תמונה מpdf
50	ניקוי המילים
51	Tokenizer
51	שימוש כל הפונקציות
52	פקנציות נספות שהספר הזה יכול לעשות
53	בניית הדטה בשביל אימון המוכנה
54	תוצאה של כל התהליכים יחד
58	ספר 2 : ספר על קוד ריצה במקביל
58	Import:
58	תהליך : יצירת תהליכונים קטנים
59	תהליכון : המרת/נקוי המילים מהקבוצ pdf ויצרת ווקטור מילים
60	יצירת תהליכים :
61	ניהול התהליכים
62	הוספת input למכונה
63	פונקציה לחישוב מקבלי
69	ספר 3 : בניית המודל של המכונה למודת
69	import
69	פונקציה ראשונה בניית המודל
70	חיזוי הנתונים
70	ריצת התוכנית לפי מה שניתן לראות כאן
73	Object Detection API Demo
73	Setup
73	Install
75	Imports
75	Model preparation

75	Variables
75	Loader
76	Loading label map
76	Detection
78	Instance Segmentation
80	Book 5: Building the AI API
80	בספר זה מדבר על בניית ה class של api
80	import
80	model traing class
80	class function
80	פנקציות לטעינת המוכנה :
81	בניית המודל של ספר 3
81	בניית המודל של ספר 3-4
81	טעינת הקבצים של
82	טעינת הפרויקט
82	מיצאת השם הקובץ
82	חזוי מסמך חדש
83	תוצאה של כל ה AI - יחד
93	בדיקה וערכה
93	ניתוך יעילות
94	אבטחת מידע
94	מסקנות על הפרויקט
94	פיתוחים עתידיים
95	בבליוגרפיה

# הצעת הפרויקט שאושרה על ידי משרד החינוך

## הסבר הפרויקט

במשרד רואי חשבון. יש מערכת שמירה קבצים ממחשבים. כדי להשתמש במערכת הזאת. עובד החברה סורק את המסמכים בתיקיות שונות. ושולח את המסמכים לשרת. אחרי שהמסמכים מגיעים לשרת העובד מתחיל לסווג את המסמכים על פי: שם הלקוח, שם החברה, שנת המסמך וסוג המסמך. אחרי סיווג המסמכים הוא שולח את הקבצים לאחראי אחסון הנתונים בחברה. ומסדר את הקבצים בתיקיות שנקבעו מראש.

## בעיות במערכת

מה עובד נדרש זמן רב להשקיע במיון הקבצים כל פעם שהלקוח מביא, חותמים, משנים את המסמך. העובד צריך לקחת את המסמך לסרוק אותו לתת לא את הפרמטרים שדורש אחראי אחסון הנתונים. והפעלות לקוחות מהעובדת זמן קריטי רב מהעובד. בנוסף על כך אחראי אחסון הנתונים צריך לשבת על המערכת הזאת זמן רב ביום בשביל לסדר אותה.

## פתרון הבעיות של המערכת

בפרויקט זה אני אבנה מערכת חדשה. אשר תקבל את הקבצים שנשלחים מהסריקה. תקראה את כל המסמך שהתקבל. ומשם היא תוציא את הנתונים הרלוונטים לכל קובץ. בזכות הנתונים שהמערכת מצאה, היא תוכל לסדר בשרת את המסמכים שהתקבלו ללא צורך שימוש העובד ואו האדם האחראי. המערכת תדע לבד את מיפוי של כל התיקיות הקריטיות בשרת ותוכל למיין את הקבצים שהתקבלו במערכת לכל תיקייה לפי הדרישות ההנהלה.

## תיאור הבעיה האלגוריתמית

כאשר הקבצים שנסרקו אין שום הבטחה שהקבצים יגיעו בצורה שלמה. לפעמים נסרקים בצורה תקולה. כמו מילים לא נקלטות או שהאותיות מתחלפות או שנימך צבע על הדף. בנוסף לך התוכן של המידע לא תמיד נמצא רק בכותרת אלה בטבלה בצעד לדוגמא. ומקומו יכול להיות שונה בכל קובץ מאותו הסוג. לפיכך קשה מאוד לבנות אלגוריתם שעובד בצורה שתטית וקבוע. על האלגוריתם חשוב לבצע. פענוח שגיאות. הבנת המידע שנסרק. ויכולת חיזוי מה יש במסמך.

## מטרות הפרויקט

- יצירת מערכת העובדת בצורת client-server
- שימוש בסיס נתונים
- שימוש בטכנולוגיית nlp
- בניית GUI (ממשק משתמש גרפי) נוח ופשוט לשימוש
- מימוש neural network
- עבודה עם servies
- שימוש פרוטוקול

## רקע תיאורטי בתחום הפרויקט

בפרויקט זה אני הולך ללמוד על הנושא של עולם ה data science תוך מימוש הכלים שלמדתי בשנתות ה יג יד. אני בחרתי בפרויקט זה להתמקד בתחום ה nlp

### nlp -

עיבוד שפה טבעית הוא תת-תחום של אינטליגנציה מלאכותית ובלשנות. הוא חוקר את הבעיות הקשורות לעיבוד ומניפולציה של שפה טבעית, והבנה של שפה טבעית על מנת לגרום למחשבים "להבין" דברים שנאמרים או נכתבים בשפות אנושיות. עיבוד השפה הטבעית קשור לתחום הבלשנות החישובית, ולעיתים משתמשים במונחים אלה ללא הבחנה ביניהם. כאן, נתייחס לעיבוד שפה טבעית כתחום המעשי של פיתוח יישומי מחשב המטפלים בשפה אנושית. הערך "בלשנות חישובית" עוסק בפן התיאורטי של שילוב רעיונות מתחום מדעי המחשב בחקר השפה האנושית.

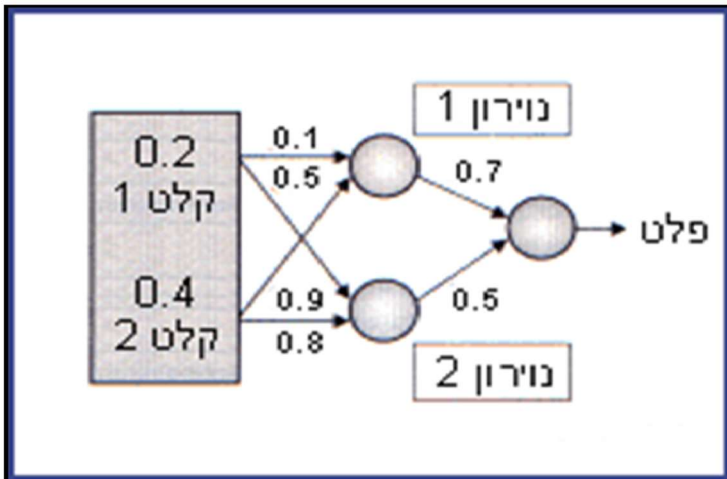
שימוש פרויקט: היות המידע שאני מקבל הם מילים ובשביל ללמד את המערכת אני צריך מספרים. אז טכניקות nlp יכול לעזור לי לתרגם את המידע המילולי למספרים וכך לאמן את המכונה

### רשת עצבית מלאכותית-

רשת עצבית מלאכותית, רשת נוירונים או רשת קשרית הוא מודל מתמטי חישובי שפותח בהשראת תהליכים מוחיים או קוגניטיביים המתרחשים ברשת עצבית טבעית ומשמש במסגרת למידת מכונה. רשת מסוג זה מכילה בדרך כלל מספר רב של יחידות מידע (קלט ופלט) הקשורות זו לזו, קשרים שלעיתים קרובות עוברים דרך יחידות מידע "חבויות" (Hidden Layer). צורת הקישור בין היחידות, המכילה מידע על חוזק הקשר, מדמה את אופן חיבור הנוירונים במוח. השימוש ברשתות עצביות מלאכותיות נפוץ בעיקר במדעים קוגניטיביים, ובמערכות תוכנה שונות - בהן: מערכות רבות של אינטליגנציה מלאכותית המבצעות משימות מגוונות - זיהוי תווים, זיהוי פנים, זיהוי כתב יד, חיזוי שוק ההון, מערכת זיהוי דיבור, זיהוי תמונה, ניתוח טקסט ועוד.

## תהליכים עיקריים בפרויקט

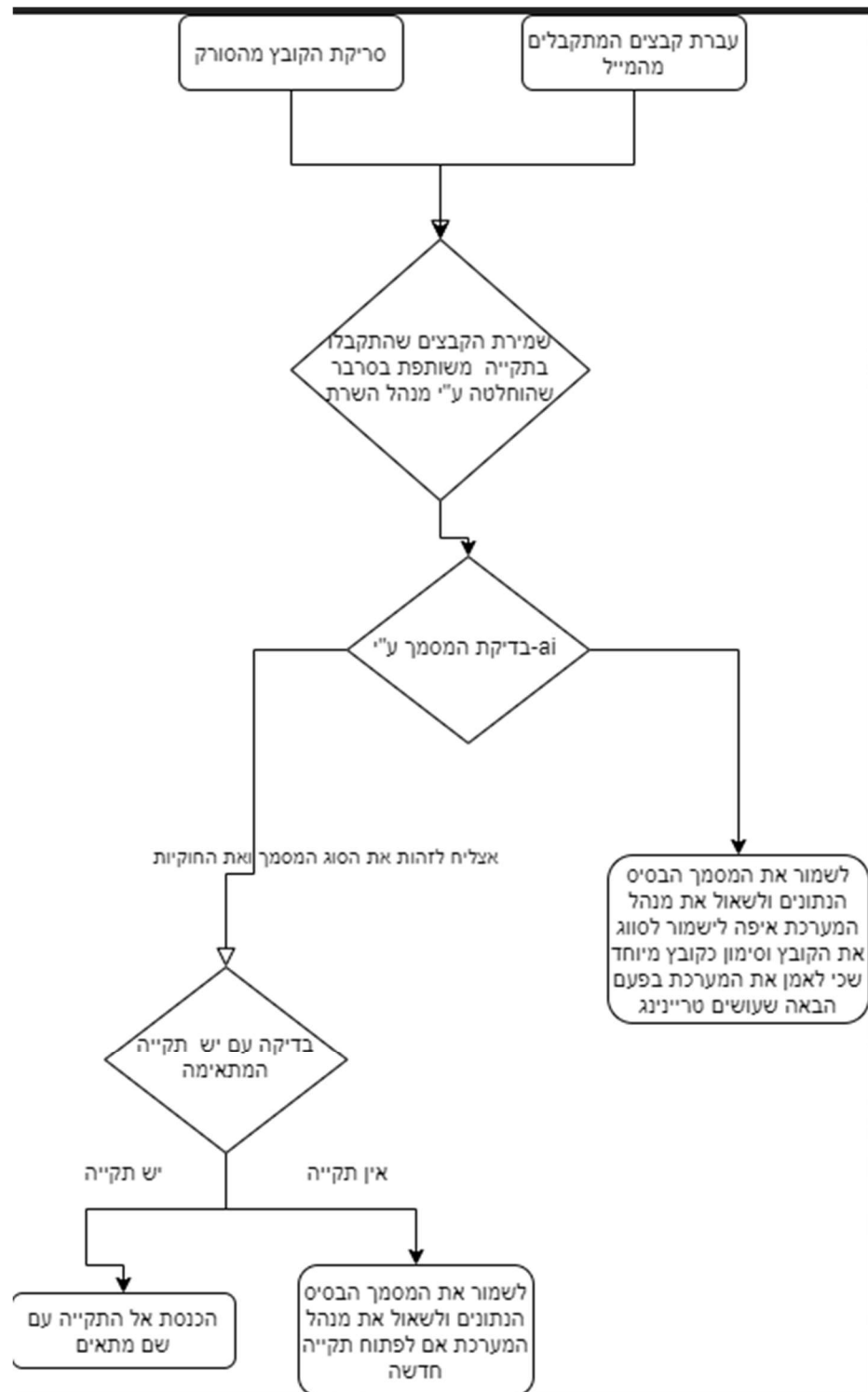
- זיהוי הקובץ המקבל.
- מעבר מתמונה למילים
- דגימת המידע המקבל תוך שימוש בMachine Learning.

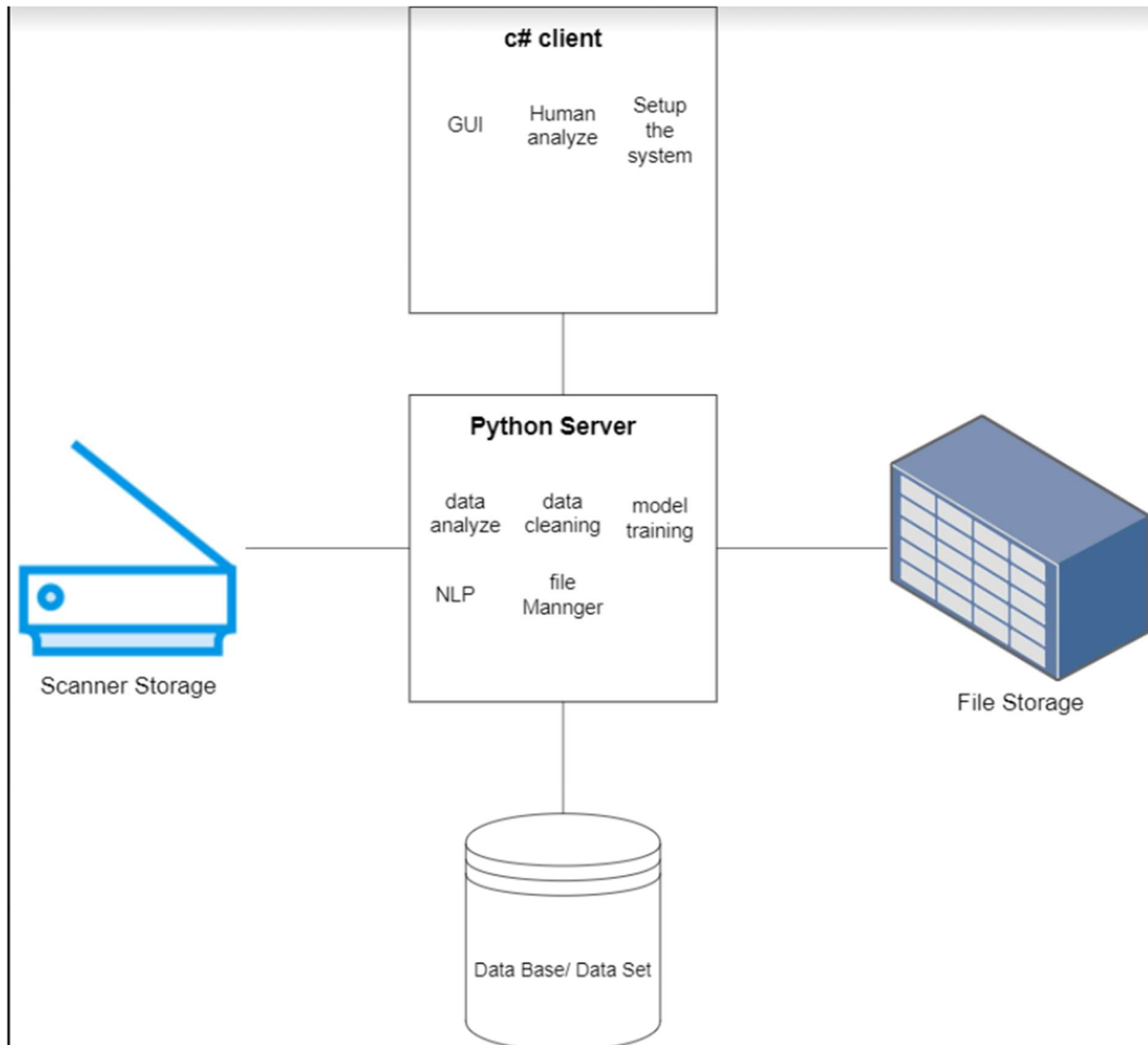


- סיווג מסמך במקום הנכון בשרת או לשלוח לעיבוד אנושי במקרה של אי ודאות



## תיאור המערכת





## צד שרת

צד שרת יכתב ב python היות התמיכה הרבה של השפה במערכת נוירונים.

- קליטת מסמכים לתוך מבנה של וקטור
- ניסיון חיזוי סוג הקובץ
- סידור קובץ לפי הדרישות שביקש הלקוח
- שליחת נתוני סיכום לצד הלקוח + מסמכים לסיווג באופן ידני

## צד לקוח

צד לקוח יכתב ב #c בעזרת טכנולוגיית ה-wpf היות ההתאמה שלה למערכת windows

- סידור המערכת לפי דרישות הלקוח
- קבלת נתונים על אופי עבודה של המערכת
- סיווג ידני של הקבצים כי שהמכונה תוכל ללמוד ממנה.

## מסד הנתונים

המסד הנתונים שהשתמש בו הוא oracle היות הזמינות שלו ליצור פונקציות מורכבות של plsql

- לאחסן נתונים של המשקלים של המערכת הניוטרוניים.
- לשמור מיקומים של קבצים שסווגו במערכת.
- שמרית העדפות לקוח במסלולי שמירה בזיכרון של השרת
- ניהול משתמשי קצה שיוכלו לגשת למערכת מצד לקוח

## דרישות עמדת פיתוח

- Visual studio
- Microsoft .NET Framework 4.5.2
- Windows מערכת הפעלה
- Python 3
- Pycharm
- Oracle Database XE

## דרישות מינימליות לעמדת המשתמש

- מערכת הפעלה : Windows 7, Windows 8.1, Windows 10
- מעבד : Intel Core 2 Quad Q9550 | AMD Phenom II X4 945 or equivalent
- זיכרון : 8 GB RAM
- כרטיס גרפי : NVIDIA GeForce GTX 1060
- DirectX: Version 9.0
- זיכרון פנוי 3GB

## פרוטוקול תקשורת

המערכת תתמוך פרוטוקול תקשורת TCP

## ביבליוגרפיה

- [NLP](#)
- [מערכת ניוטרוניים](#)

## לוחות זמנים

שלב		טווח תאריכים	
		מ-	עד-
למידת חומר תאורטי		1/9/2020	10/11/2020
בניית מערכת נירונים		10/11/2020	30/12/2020
פיתוח התוכנה/קידוד		1/1/2021	30/3/2021
ניפוי השגיאות		1/4/2021	15/4/2021
הרצת ניסיון		16/4/2021	30/4/2021
הגשת הפרוייקט		5/2021	5/2021

## תקציר / מבוא

### רקע הפרויקט

במשרד רואי חשבון. יש מערכת שמירה קבצים ממחשבים. כדי להשתמש במערכת הזאת. עובד החברה סורק את המסמכים בתיקיות שונות. ושולח את המסמכים לשרת. אחרי שהמסמכים מגיעים לשרת העובד מתחיל לסווג את המסמכים על פי: שם הלקוח, שם החברה, שנת המסמך וסוג המסמך. אחרי סיווג המסמכים הוא שולח את הקבצים לאחראי אחסון הנתונים בחברה. ומסדר את הקבצים בתיקיות שנקבעו מראש. מה עובד נדרש זמן רב להשקיע במיון הקבצים כל פעם שהלקוח מביא, חותמים, משנים את המסמך. העובד צריך לקחת את המסמך לסרוק אותו לתת לא את הפרמטרים שדורש אחראי אחסון הנתונים. והפעלות לקוחות מהעובדת זמן קריטי רב מהעובד. בנוסף על כך אחראי אחסון הנתונים צריך לשבת על המערכת הזאת זמן רב ביום בשביל לסדר אותה.

### תהליך המחקר

כאשר הקבצים שנסרקו אין שום הבטחה שהקבצים יגיעו בצורה שלמה. לפעמים נסרקים בצורה תקולה. כמו מילים לא נקלטות או שהאותיות מתחלפות או שנימך צבע על הדף. בנוסף לך התוכן של המידע לא תמיד נמצא רק בכותרת אלה בטבלה בצעד לדוגמא. ומקומו יכול להיות שונה בכל קובץ מאותו הסוג. אז לכן לפני שנתחיל לבנות את המערכת, נהיה צריכים לבדוק אם יש אפשרות לקבל מסמך pdf את מידע הכתוב ממנו במילים. אשר מתארות את הקובץ ולכן הוא שייך. ולהציג אותם כפלט לסביבת עבודה המיועדת לתכנות. בנוסף לקח מסביבת העבודה צריכה לתמוך בשפות שונות (כגון עברית) היות שהסמיכים שאנו מקבלים לא חייבים להיות הכרח אנגלית שזאת השפה המקובלת לכתיבת קוד ולפרייקטים. בעיות האלה מצאנו פתרונות והחלטנו לפתח בסביבת העבודה של python, המאפשרת צרות קידוד של utf-8. בזכות הקידוד הזה ניתן לעבוד בשפות אחרות כמו עברית. המרת קבצים לפלט לסביבה העבודה יש את ה-ocr של גוגל אשר יודע לקלוט מילים בתמונות ומסמכים שונים. ועל ה-ocr אני אפרט בהמשך.

### סקירה ספרות

עיבוד שפה טבעית הוא תת-תחום של אינטליגנציה מלאכותית ובלשנות. הוא חוקר את הבעיות - [nlp](#) הקשורות לעיבוד ומניפולציה של שפה טבעית, והבנה של שפה טבעית על מנת לגרום למחשבים "להבין" דברים שנאמרים או נכתבים בשפות אנושיות.

רשת נוירונים או רשת קשרית הוא מודל מתמטי חישובי שפותח בהשראת - [רשת עצבית מלאכותית](#) המתרחשים ברשת עצבית טבעית ומשמש במסגרת למידת מכונה תהליכים מוחיים או קוגניטיביים.

[Google OCR](#) - Optical Character Recognition (OCR) The Vision API can detect and extract text from images. There are two annotation features that support optical character recognition (OCR)

[Multiprocessing](#) - Multiprocessing refers to the ability of a system to support more than one processor at the same time. Applications in a multiprocessing system are broken to smaller routines that run independently. The operating system allocates these threads to the processors improving performance of the system.

## אתגרים מרכזיים

### הבעיה איתה מתמודד התלמיד

במהלך הפרויקט היינו צריכים להתמודד עם בעיה אימון המוכנה. כדי לאמן מכונה יש שני דרישות שצריכים לענות עליהם כדי שהמכונה תוכל לבצע את המשימה הוגדרה באופן אמין. והמשימות הללו היו:

### הנגשת המידע למכונה

הפלט שאנו מקבלים מהמסמך הוא נכתב במילים שאדם יכול להבין כמו עברית אנגלית. אבל המחשב מבין רק את השפה הבינארית שהיא 0 או 1. כך נוצר בעיה של לימוד המכונה הלומדת שלנו. אנו צריכים המכונה הלומדת תאבחן את המשמעות של כל מילה, ותיגש אליה בצורה אופטימלית ביותר כדי לפתור את הבעיה כדי לפתור את הבעיה. אז אנו צריכים לבנות מודל מתמטי, שיאפשר לנו לגשר בין הפער של השפה שלנו ושפת המחשב. שיכול לזהות את משמעותו של מסמך, ותוכל לספק לנו את הנתונים הדרושים

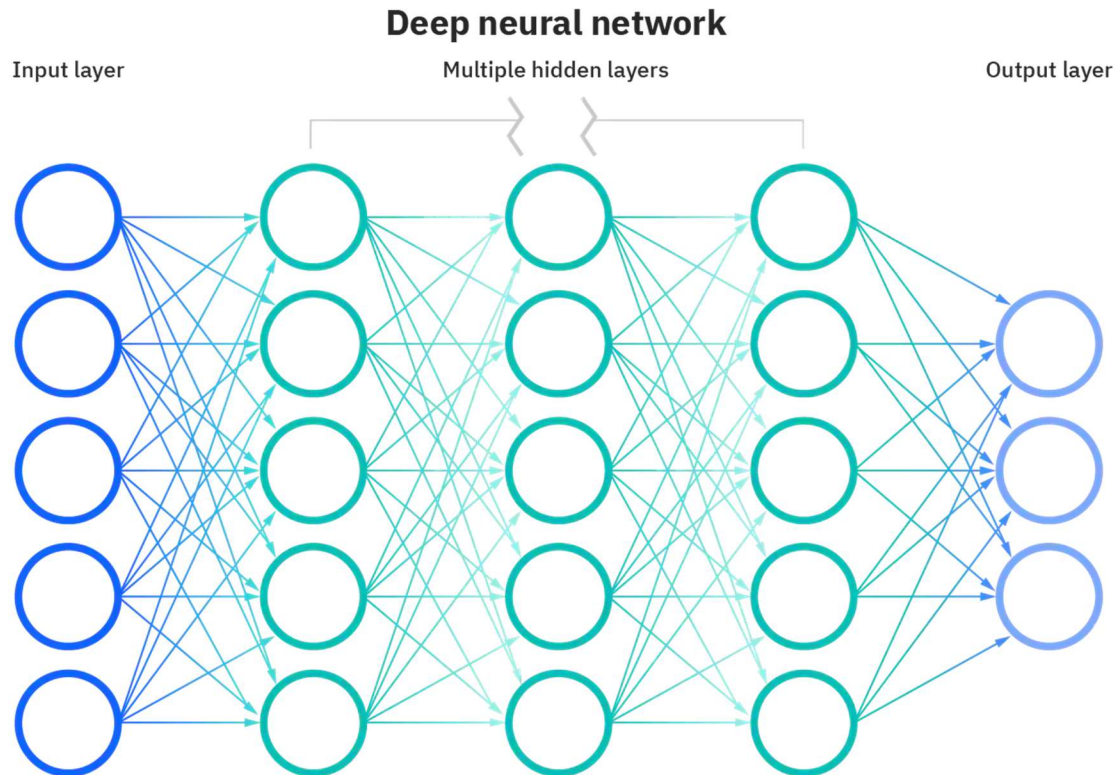
### ניקוי המידע

אחת מהדרישות היא קבלת פלט נקי למכונה הלומד. משמע אנו צריכים לסנן מידע ומילים לא רלוונטיות אשר יורידו את האמינות של המכונה. כלומר אם מקבל טופס ללמוד המוכנה. הטופס שהתקבל ללמוד מוכנה הוא בעייתי (מחצית מהמילים נמחקו, הסריקה לא הוצלחה). ומה שנקבל זה גיבריש (כמו מילים: צראטפף, עיאה או סתם צירופים של אותיות בודדות ללא שום קשר) נלמדות מספר רב של פעמים, תוצאה מכך המכונה תתן חשיבות גם לערכי הגיבריש. כך האמינות של המוכנה תפגע באופן משמעותי. בנוסף לקח המידע שאנו רוצים לקבל צריך להיות באותו פורמט. למשל עם הטקסט הכתוב במסמך הוא בעברית. אז את כל המילים הלועזיות צריך לתרגם לעברית, או יש לנו המון מספרים מייצגים תאריכים שונים. אנו צריכים לאחד את כל הסוגים של המילים האלה לסוג אחד מרכזי שממנו נוכל לאמן את המערכת.













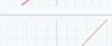
### בניית המוכנה

אחד מהבעיות המרכזיות היא בניית המודל שאנו רוצים להכין. בניית רשת נוירונים היא מורכבת מאוד צריך לקחת המון דברים בחשבון משפיעה על הצלחתה של המערכת. למשל:

- מספר שכבות hidden רשת נוירונים מורכבת מלפחות שלושה שכבות. שכבת ה input אשר האחרית לקבל את הקלט של המכונה שתוכל לבצע את השלב הבא. שכבה אחרונה היא שכבת output שהיא שיכבה באחרית להגיד באופן הסתברותי מה המוכנה חושבת שזה הפלט. ויש n שיכבות hidden. שכבות hidden בס שכבות המכילות מספר נוירונים שבהם יש את הפרמטרים לזיהוי המסמך. בעיקרון שכבות hidden עושות מניפולציות מתמטיות שונות כדי להחליט מה יהיה ה output. תוצאה ממבנה זה אחד משאלות החשובות ביותר ביא כמה שכבות hidden צריך לעשות. כמה נוירונים (פרמטרים) צריך בכל שכבה.



- פרמטר נוסף שמאוד חשוב בבניית המוכנה Activation function. מטרת Activation function היא מציאת הפרמטרים המתאים בשביל כל נוירון ברשת נוירונים. ניתן לתאר את העובדה של Activation function כך. ה-Activation function היא המדריך שלנו למצוא את הנקודה הכי נמוכה על הר מסוים כאשר העיניים שלך סגורות. כלומר אתה נמצא על הר מסוים. שהמטרה שלך היא לרדת מהר לנקודה הכי נמוכה שקיימת באזור זה. ה-Activation function היא בעצם המדריך שלנו להגיע לנקודה הכי נמוכה לפי הדוגמא הזאת. אבל במציאות ה-Activation function עושה את זה ב-n מימדים. ויש המון Activation function שמפעלות בצורה שונה.

Name	Plot	Function, $f(x)$	Derivative of $f$ , $f'(x)$	Range	Order of continuity
Identity		$x$	1	$(-\infty, \infty)$	$C^\infty$
Binary step		$\begin{cases} 0 & \text{if } x < 0 \\ 1 & \text{if } x \geq 0 \end{cases}$	$\begin{cases} 0 & \text{if } x \neq 0 \\ \text{undefined} & \text{if } x = 0 \end{cases}$	$\{0, 1\}$	$C^{-1}$
Logistic, sigmoid, or soft step		$\sigma(x) = \frac{1}{1 + e^{-x}}$ <sup>[1]</sup>	$f(x)(1 - f(x))$	$(0, 1)$	$C^\infty$
tanh		$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$	$1 - f(x)^2$	$(-1, 1)$	$C^\infty$
Rectified linear unit (ReLU) <sup>[7]</sup>		$\begin{cases} 0 & \text{if } x \leq 0 \\ x & \text{if } x > 0 \end{cases}$ $= \max\{0, x\} = x \mathbf{1}_{x>0}$	$\begin{cases} 0 & \text{if } x < 0 \\ 1 & \text{if } x > 0 \\ \text{undefined} & \text{if } x = 0 \end{cases}$	$[0, \infty)$	$C^0$
Gaussian Error Linear Unit (GELU) <sup>[4]</sup>		$\frac{1}{2}x \left( 1 + \operatorname{erf}\left(\frac{x}{\sqrt{2}}\right) \right)$ $= x\Phi(x)$	$\Phi(x) + x\phi(x)$	$(-0.17 \dots, \infty)$	$C^\infty$
Softplus <sup>[8]</sup>		$\ln(1 + e^x)$	$\frac{1}{1 + e^{-x}}$	$(0, \infty)$	$C^\infty$
Exponential linear unit (ELU) <sup>[9]</sup>		$\begin{cases} \alpha(e^x - 1) & \text{if } x \leq 0 \\ x & \text{if } x > 0 \end{cases}$ with parameter $\alpha$	$\begin{cases} \alpha e^x & \text{if } x < 0 \\ 1 & \text{if } x > 0 \\ 1 & \text{if } x = 0 \text{ and } \alpha = 1 \end{cases}$	$(-\alpha, \infty)$	$\begin{cases} C^1 & \text{if } \alpha = 1 \\ C^0 & \text{otherwise} \end{cases}$
Scaled exponential linear unit (SELU) <sup>[10]</sup>		$\lambda \begin{cases} \alpha(e^x - 1) & \text{if } x < 0 \\ x & \text{if } x \geq 0 \end{cases}$ with parameters $\lambda = 1.0507$ and $\alpha = 1.67326$	$\lambda \begin{cases} \alpha e^x & \text{if } x < 0 \\ 1 & \text{if } x \geq 0 \end{cases}$	$(-\lambda\alpha, \infty)$	$C^0$
Leaky rectified linear unit (Leaky ReLU) <sup>[11]</sup>		$\begin{cases} 0.01x & \text{if } x < 0 \\ x & \text{if } x \geq 0 \end{cases}$	$\begin{cases} 0.01 & \text{if } x < 0 \\ 1 & \text{if } x \geq 0 \end{cases}$	$(-\infty, \infty)$	$C^0$
Parametric rectified linear unit (PReLU) <sup>[12]</sup>		$\begin{cases} \alpha x & \text{if } x < 0 \\ x & \text{if } x \geq 0 \end{cases}$ with parameter $\alpha$	$\begin{cases} \alpha & \text{if } x < 0 \\ 1 & \text{if } x \geq 0 \end{cases}$	$(-\infty, \infty)^{[2]}$	$C^0$
Sigmoid linear unit (SiLU) <sup>[4]</sup> Sigmoid shrinkage, <sup>[13]</sup> SiL, <sup>[14]</sup> or Swish-1 <sup>[15]</sup>		$\frac{x}{1 + e^{-x}}$	$\frac{1 + e^{-x} + xe^{-x}}{(1 + e^{-x})^2}$	$[-0.278 \dots, \infty)$	$C^\infty$
Gaussian		$e^{-x^2}$	$-2xe^{-x^2}$	$(0, 1]$	$C^\infty$

## הסיבה לבחירת הנושא

כיום אנו נמצאים בעידן שכל הדברים החשובים שלנו נמצאים במחשב. למשל הפגישות שלנו עם חברים או עובדים מהמשרד. בזכות הטכנולוגיה המידע הוא מאוד נגיש. אבל עדיין יש המון מקצועות משרדים ישנים. שעובדים בשיטה הישנה של קלסרים. ולכן הם צריכים לעשות שינוי בגישה הטכנולוגית שלהם. לכן הם צריכים לעבור על המון המון מסמכים שונים ולהקליל את הנתונים באופן עצמאי. התהליך להעביר כמות גדולה של מסמכים לוקחת המון זמן ולכן היא עולה המון כסף. ובמקום שעובד תעסק בעבודה שלו הוא צריך להתעסק בסקירת המסמכים ולסדר אותם. לכן פרויקט זה יכול לחסוך המון זמן וכסף.

## מוטיבציה לעבודה

בעבודה זאת החלטנו לנסות ללמוד מהי מכונה לומדת. לנסות ולהבין אל אילו שימושים האלגוריתמים היא עונה. כדי להבין מה ואיך עובד העולם של AI לנסות לראות איך נוכל להשתלב אליו. היות שהפרויקט הזה הוא פשוט בבחינת הדרישות שלו, לעומת לימוד מכונת לנסוע בדרך עירונית. לכן נוכל ללמוד על נושא זה בצורה ממוקדת ועניינית. חוץ מי זה את הפרויקט זה נעשה ביחד שני אנשים. וזאת תהיה פעם ראשונה של שנינו על עבודה בקבוצה וכתובת קוד של יותר מאדם אחד.



## הצרכים שהפרויקט עונה אליהם

פרויקט זה נותן גישה מהירה בסדר וארגון של מידע. בזכות פרויקט מסוג זה. יהיה יותר קל להעביר מסמכים מדף למחשב. בזכות זאת צורת העבודה של חברות שונות תוכל להתנהל בצורה יעילה יותר. וכך להספיק יותר ולשמור על סדר העבודה קבוע. עם יכולת לדעת הקבצים נשמרים במחשב בצורה מאורגנת ומסודרת לפי איך נקבע. משום העובדה שאנו משתמשים במכונה לומדת יהיה יותר קל להוסיף עוד סוגים של מסמכים לעתיד וכך להגדיל את מאגר הקבצים שיש במערכת. יהיה קל לחברות שונות להוסיף מסמכים שונים ולהגיד את צורת הארגון שלהם.

## הפתרון לבעיה יצירת המידע

כדי לפתור את בעיית התרגום הפלט נשתמש בטכניקות nlp. nlp הוא חוקר את הבעיות הקשורות לעיבוד ומניפולציה של שפה טבעית, והבנה של שפה טבעית על מנת לגרום למחשבים "להבין" דברים שנאמרים או נכתבים בשפות אנושיות. עיבוד השפה הטבעית קשור לתחום הבלשנות החישובית, ולעיתים משתמשים במונחים אלה ללא הבחנה ביניהם. כאן, נתייחס לעיבוד שפה טבעית כתחום המעשי של פיתוח יישומי מחשב המטפלים בשפה אנושית. הערך "בלשנות חישובית" עוסק בכך התיאורטי של שילוב רעיונות מתחום מדעי המחשב בחקר השפה האנושית. ל nlp יש מודל מיוחד שיכול לעזור לנו לפתור את בעיית התרגום והוא נקרא bag of word. שהוא לוקח את כל המילים המופיעות בעולם. ללא שום קשר לצורת המילה או לכללי הדקדוק שלה. ומכל המילים המופיעות ב bag of words ביחד מייצגים את העולם המילים שלנו. ואנו נתייחס רק למילים המופיעות שם.

Document	the	cat	sat	in	hat	with
the cat sat	1	1	1	0	0	0
the cat sat in the hat	2	1	1	1	1	0
the cat with the hat	2	1	0	0	1	1

לפי התמונה שאנו רואים כאן בתמונה. שהמילים שיש לנו בתוך bag of word הם:

the, cat, sat, in, hat, with.

ואחרי זה נעבור משפט משפט ונסמן כמה פעמים המילה מופיע בטקסט הזה. לפי הדוגמא במשפט the cat sat. אנו נעבור על כל מילה ומילה ונכתוב כמה פעמים היא מופיע. ניתן לראות שבדוגמא זאת. the מופיע פעם אחת cat מופיע גם פעם אחת. sat גם מופיע פעם אחת. אבל המילים in hat with לא מופיעות בכלל בטקסט המודבר. ולכן המילים האלה יסמנו כלא קיימים בטקסט.

בזכות היכולת של מודל זה נוכל לקבוע את הקלט שהוכנס למכונה. אנו נשתמש במודל הזה כדי ליצור מבנה קבוע אחיד שממנו נאמן את המכונה שלנו. לפי מה שידוע לנו על אימון מכונה אנו צריכים לארגן כמו גדולה מאוד של מסמכים מאותו סוג רק קובץ שונה. בהתחלה. אנו נעבוד על כל אחד מהמסמכים ונעבור על המילים שלו. אנו נוסיף לתוך ה bag of word את כל המילים הקיימים במסמך. ואת זה נעשה לכל המסמכים שאנו

רוצים שיהיו במערכת. נעבוד בנה נתונים של dictionary וכך לכל אות יש ערך מספרי.

wordfreq - Dictionary (535 elements)

Key	Type	Size	Value
a	int	1	31
in	int	1	28
to	int	1	27
language	int	1	25
and	int	1	24
natural	int	1	20
machine	int	1	16
processing	int	1	16
systems	int	1	15

Save and Close

Close

אחר כך ניקח את כל המילים שיש המילון ונבנה וקטור (מערך מאוד גדול) שאינדקס הראשון יהיה את המילה השכיחה ביותר. ובמקום האחרון יהיה את המילה הנדירה ביותר. לאחר מי כאן עברנו על הוקטור והסתכלנו מאיפה המילים הופכים לגיבריש. היות הגיבריש משתנה כל הזמן ואין לא צרות קבועות. אז ידוע לנו שהוא תמיד נמצא באינדקסים האחרונים של הוקטור. אז חתכנו את הוקטור לגודל יחיד שבאו אין גיבריש. והחלטנו שזה יהיה התבנית שלנו של הקלט שנכנס לתוך המכונה. את התבנית ניקח לכל מסך כך שנעבור מסמך מסמך נסמן עם המילה קיים או לא. ונמסמן בתבנית 1 אם המילה קיים יותר מפעם אחת אחרת נסמן באפס עם היא לא קיימת. אז מה שיקרה נקבל מבנה קבוע לכל המסמכים. שזכותו נוכל לאמן את מכונה.

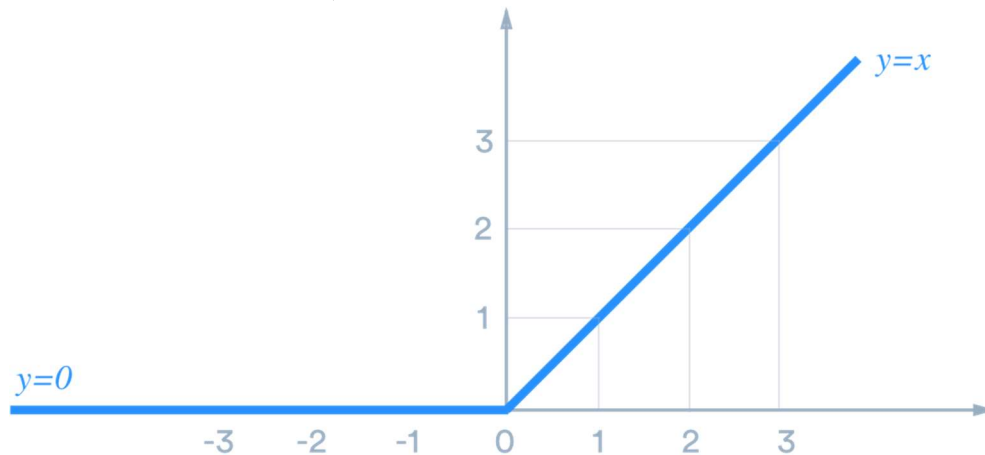
	0	1	2	3	4	5	6	7	8	9	10	11
0	1	1	1	1	1	1	1	1	1	0	0	0
1	0	0	0	1	0	1	1	1	1	0	0	0
2	1	1	0	1	0	1	0	1	1	0	0	0
3	1	1	1	1	0	0	1	0	0	0	0	0
4	1	1	0	1	0	0	0	0	0	0	0	0
5	1	0	1	0	0	0	0	0	0	1	1	0
6	1	0	0	1	1	0	1	0	0	1	1	0
7	1	0	0	1	0	0	0	0	0	1	0	1
8	1	1	1	1	0	1	1	1	1	0	0	1

לפי מה שניתן לראות בתמונה הזאת המילה הראשונה נמצאה במסמך שלקחנו. וגם השמונה המילים אחריו הכי נפוצים. אבל המילה העשירית בכלל לא מופיע במסמך לכן היא מסומנת באפס. בזכות צורת המודל הזאת לייחד מסמך מסויים יש רצף שונה של אפס או אחד. ועם זה המוכנה צריכה להתמודד. וכל נירון במערכת צריך למצוא משקל שונה כדי להבחין בין סוג שונה של תופס.

## הפתרון לבעיה יצירת המודל

את פתרון בניית המודל של המכונה פתרנו אותה בספר שלבים אשר יצרו לנו את המודל המתאים למערכת:

- שכבת ה input: בזכות התהליך להכנסת המידע שלנו. קיבלנו וקטור חד מימדי אשר מכיל את כל המילים של הקבצים. לכן החלטנו בשכבה הזאת כל ערך בווקטור הוא נורמון מפני עצמו ונתייחס מערך הזה בתור מערך נורמונים
- כמות שכבות hidden: היות שהמערכת שלנו נוראה פשוטה ותהליך ה nlp יחסוך לנו המון שלבים בעיבוד המידע. אז החלטנו שלא נבנה במערכת deep learning (מערכת עם כמה שכבות hidden) בזכות המערכת של ה nlp המתוכננת היטב.
- Activation function: את ה Activation function הראשון שרצינו להשתמש באו הוא: relu



בחרנו פונקציה הזאת היות וזמן החישוב שלה יותר מהיר. בזכות העובדה שהיא משתמשת ב

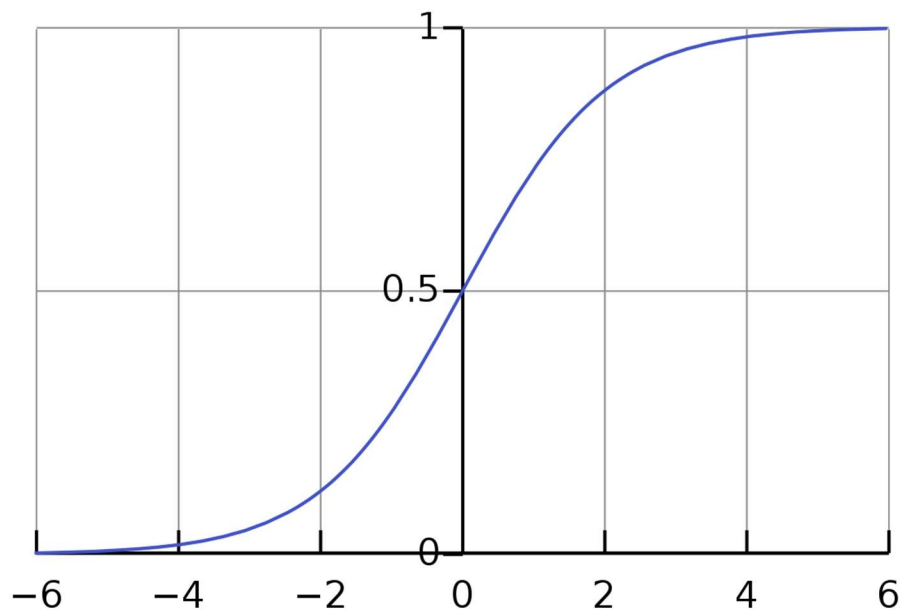
$$f(x) = \begin{cases} x & \text{if } x > 0, \\ 0.01x & \text{otherwise.} \end{cases}$$

וזמן עיבוד למחשב לוקחת יותר מהר מאשר פונקציות שעובדות בצורה מערכית. בנוסף לקח השתמשו בא היות שהיא עובדת בטווח של 0 או 1. זה הטווח שהוגדר למערכת של nlp. אבל בפועל כאשר ביצענו את בניית המכונה בפונקציה שנתונה לנו את התוצאות הטובות ביותר.

$$f(x) = \frac{1}{1 + e^{-x}}$$

היא פונקצית

אשר נראית בפועל כך.



- את כמות הניורונים בשכבת hidden יתבצע על ידי ניסוי וטעייה. עד שמצאנו המספר המתאים בשכבת hidden הוא 16
- בשכיבת output הגדרנו שני ניורונים. הסיבה לכך היא שכבה הזאת התוצאה שאנו מקבלים זה התאמה לכל סוג מסמך. כל ניורון בודק את ההתאמה לכל סוג מסמך.

## מטרות ויעדים ומדדים להצלחה

### מטרות

- לזהות מסמך באופן אוטומטי ולהוציא את הנתונים הרלוונטים מהמסמך.
- להצליח לגרום למערכת לעבוד לבד. ללא תלות באדם. על אחוז גובה של תהליכי העבודה.
- לנהל מערכת אמינה וחדשנית שתטפל כמות מסמכים שונים ותוכל לעבוד עם משתמשי קצה שונים.
- לתת פתרון נוח לסיווג מסמכים למשתמשי הקצה של המערכת. בשביל ניהול יותר מוסדר.

### יעדים

- לבנות מודל מתמטי לבעיה, שנוכל לממש אותו בקוד.
- לבנות מודל של רשת נוירונים אשר תוכל לזהות מסמכים שונים.
- לימוד המוכנה את כל מסמכי המערכת בצורה יעילה.
- בניית GUI נוח למשתמש, שהוא יוכל לצפות בנתונים של התוכנה בצורה ברורה לקוח.
- מניית משתמשים זרים משימוש במערכת. פגיעה ביכולת לתפקד.



## אתגרים

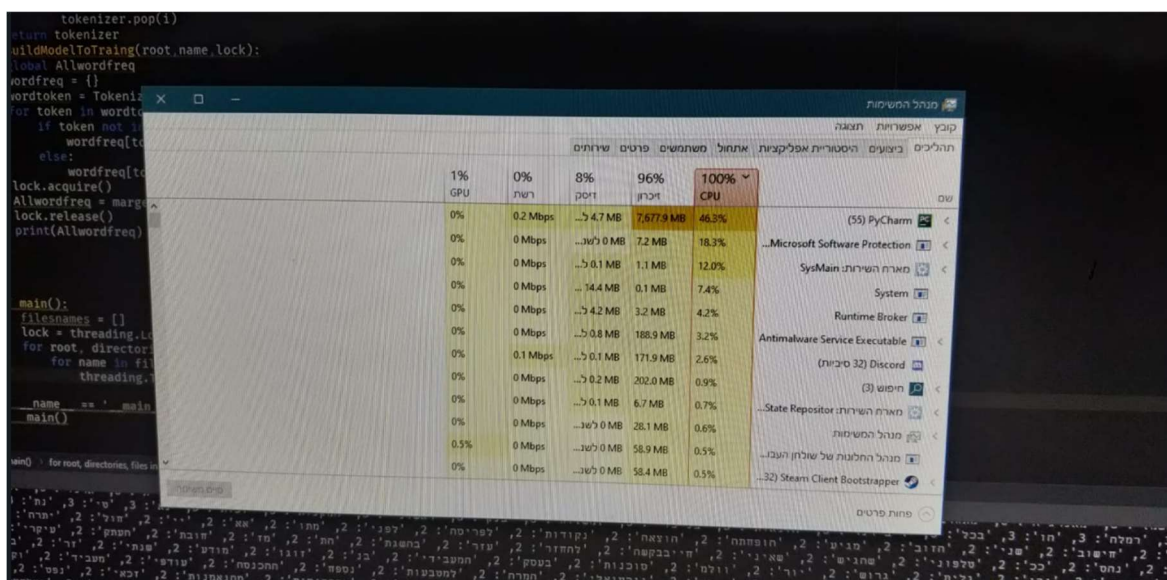
בבניית הפרויקט זה נתקלנו בבעיות שונות בזמן הפיתוח. הבעיות האלה היו:

- אחד האתגרים המרכזיים שהייתה לנו היא אימון המסמכים. הקוד שמחלץ את המילים מן המסמך הוא קוד מאוד כבד. הקוד הזה פונה בכמות גדולה מאוד api שונים, בשביל לקבל מידע חיצוני. כמו איך לתרגם מילים באנגלית או תרגום מספר למילה (מספר 1000 המערכת צריכה להכיר אותו כי "אלף" או צירופים מאוד מורכבים, גם המערכת צריכה לבצע כמו 1243. הערכת צריכה לתרגם את המספר זה ל- " אלף מאתיים ארבעים ושלוש). והזמן שלוקח למערכת לעבוד משלב הוצאת המידע מן מסמך. ועד הכנת וקטור מילים של המסמך, לוקחת מספר דקות רבות בשביל מסמך אחד. למרות העובדה שמערכת שלנו מתוכננת לעבוד בתור תהליך אצווה (offline), כדי לאמן את המערכת שלנו אנו זקוקים לדרך מהירה לעשות את התהליך הזה. סביה לקח היא שיש לנו 250 מסמכים לאמן. ועם מסמך אחד לוקח לאמן 2 דקות. 250 מסמכים לוקח כמעט יממה. ותחילת התהליך אנו זקוקים לעשות את הפעולה זאת מספר רב של פעמים. היות שינוי הערכים בוקטור, הוספת סוג מסמך חדש שהמערכת צריכה להכיר, שינוי טכנולוגית עבודה וכ"ו. בנוסף לקח אנו לא רוצים שהלקוחות אשר משתמשים במערכת יצטרכו לחכות יממה שלמה בשביל שהקבצים שלהם ימינו והגיעו למקום הנכון. לדוגמא: אם בחברה רוצים למיין אלף מסמכים יקח להם כמעט שבוע. לכן אנו צריכים לציור במערכת שלנו אפשרות לעשות חישובים במקביל לעשות את פעולת יצירת וקטור המילים במקביל. באופן יעיל שלא ישפיע באמינות וביעילות המערכת.
- אתגר נוסף שנתקלנו במהלך הכנת הפרויקט. היה הוצאת נתונים מהקובץ. לאחר שגילנו מה המסמך שלנו, ואיזה נתונים חשובים יש להוציא ממנו. אנו זקוקים לחפש את הנתונים האלה ושומר אותם כדי לשלוח ללקוח. לדוגמא בכל מסמך יש לנו שם לקוח. ונתון זה הוא נתון מאוד קריטי למערכת שלנו. כי מרכיב המרכזי לסידור מסמכים הוא השמות של הלקוחות. אנו צריכים לדעת בכל מסמך לזהות את השם של הלקוח, ולדעת את המיקום המדויק שהשם הזה נמצא. כלומר יש מסמכים בהם מופיע יותר משם אחד. לדוגמא ליד שם הלקוח יש את שם אביו ואת בן או בת זוגו של הלקוח. יש מצבים שבכלל אנו לא מחפשים שמות אנשים. כמו שם חברה או שותפות. לכן אנו צריכים להבדיל בין השמות המסמכים. וכל מסמך מאותו סוג יש קומבינציה שונה. אז עלינו ליצור שיטה שתוכל לזהות בכל מסמך באופן אוטומטי את המיקום של הנתונים כמו שהצגתי פה את שם הלקוח. אשר תסמן לתוצאי את המיקום של המסך. מימוש הפתרון היה בעזרת מודל של tensorflow, אשר ממש זיהוי באובייקטים. tensorflow יש מודל מכונה בשם faster rcnn. אשר מאפשר ללמד בקלות את המודל. ומסמן את החלק שלמדנו כתוצאה. אנו למדנו את מודל זה את כל המידע שאנחנו צריכים להוציא.





- אתגר נוסף שהיה לנו הוא עמוס בתפעול המערכת. באחד מהפתרון של האתגר הראשון שכתבו. יצרנו מערכת לחישוב במקביל המבוססת על threading. אבל במהלך יצירת המערכת. נתקלנו בבעית עומס רציני על ה cpu וכל זיכרון ה ram של מחשב.



לכן אנו זקוקים לכך שמערכת. תנהל את התהליכים שלה בצורה יותר יעילה. בכך שפצל את התהליך הראשי לכמה תהליכים קטנים. שכל תהליך יחזיק עוד תהליכים קטנים. ואת החשובים הכבדים ניתן לטען לעשות אותם בשביל לחסוך בזמן.

## מדדי הצלחה למערכת

המערכת מצליחה לסיווג מסמכים באופן אוטומטי. ואת המסמכים המערכת שלא הצליחה לזהות היא שולחת לממשק ה gui של משתמש קצה.

## רקע תיאורטי

התחום שבו נתעסק הוא למידת מוכנה בעזרת רשת נוירונים. את הנושא שנחקר הוא זיהוי אובייקטים. בעזרת למידה מראש. אנו נשתמש במחקרים. ההרצאה של אוניברסיטת סטנפורד<sup>1</sup>. אשר מציג את אופן אימון התאוריתי של המכונה, איך עובד רשת נוירונים. את תרגום שפת האדם לשפת מכונה בעזרת תכנות nlp נשתמש במחקר של אליס זהון<sup>2</sup>. את בניית מודל שנכניס אותו בתור input למכונה ניעזר בכתבה של Usman Malik<sup>3</sup>. בשביל לבנות את מודל המוכנה אשר מזהה את סוג המסמך. נשתמש בסרטוני הדרכה של גוגל<sup>4</sup>. בנוסף לקח אנו רוצים לזהות מסמכים מתמונה. אז git של EdjeElectronics<sup>5</sup> נבנה מכונה אשר מזהה חלקים קריטיים בסמכים. לסיכום בשביל הפרויקט זה. חקרנו את ההתנהגות של המערכת נוירונים. ולממש את המערכת התאוריה הזאת, למערכת פרקטית עובדת. אנו רואים שביל ליצור את המערכת שלנו, ניצטרך מערכת עיבוד נתונים חזקה. אשר תוכל לחשב המון נתונים בזמן סביר.

## תיאור מצב קיים

כיום העברה של המסמכים מתבצע בצורה ידנית. יש אדם אשר סורק את המסמכים באופן ידני. אחרי שהוא סורק אותם הוא עובר אחד אחד על המסמך ובודק. אם סריקה תקינה, איזה סוג מסמך זה. ואת שם הלקוח שמופיעה המסמך. אחרי שלב זה האדם מחפש את הלקוח בשרת של החברה. ומסווג את המסמך למקום המתאים בשרת. המסמכים שנסרקים בסורק הם דרך כלל קובעים. ולתאים נדירות יש סריקה אשר לא קשורה לאף מסמך. ואת סריקת הזבל צריך למחוק את המסמך. מה שנדרש מאיתנו זה לחפש סכום מכובד של מסמכים מאותו סוג לכל סוגי הסמכים במערכת. ללמוד איזה לקוח יש בחברה. ומה המיקום של כל לקוח בשרת. וכך נוכל לבנות מערכת אוטומטית אשר תסווג את המסמכים ללא תלות באדם.

## ניתוח חלופי למערכת

חסרונות	יתרונות	תיאור המערכת
<ul style="list-style-type: none"> <li>המערכת מאוד איטית.</li> <li>צריך כוח עיבוד חזק בשביל להחזיק את המערכת</li> <li>צריך הרבה מאוד זמן</li> </ul>	<ul style="list-style-type: none"> <li>התוצאות אמינות יותר</li> <li>המערכת יכולה להשתפר ככל שהזמן עובר</li> <li>המערכת גנרית</li> </ul>	<p>יצירת מערכת לומדת. אשר תלמד את סוגי המסמכים. ותבצע סיווג מסמכים על פי מה שהיא למדה</p>

<sup>1</sup> [קישור לקורס המלאה של אוניברסיטת סטנפורד](#)

<sup>2</sup> [קישור למחקר של אליס זהון](#)

<sup>3</sup> [קישור לכתבה של Usman Malik](#)

<sup>4</sup> [קישור לסרטוני הדרכה של גוגל](#)

<sup>5</sup> [קישור ל git של EdjeElectronics](#)



<ul style="list-style-type: none"> <li>● פתרון חדשני ומתוחכם</li> </ul>	<ul style="list-style-type: none"> <li>● בשביל ללמוד איך ליצור מערכת הזאת</li> <li>● קשה יותר להוסיף מסמכים חדשים</li> </ul>	
<ul style="list-style-type: none"> <li>● קל להכנה</li> <li>● קל להוסיף מסמך חדש למערכת ללא שום מאמץ נוסף</li> </ul>	<ul style="list-style-type: none"> <li>● המערכת לא תהיה אמינה מספיק</li> <li>● לא יכולה לעבוד טוב עם מסמכים שקצת שונים</li> <li>● מערכת סטטית לכל template ולא תוכל לעמוד בשינוי מהמסמך</li> <li>● המערכת טיפשית</li> </ul>	<p>יצירת מסמך template לכל סוג של קובץ ולבדוק בכל קובץ כמה מילים מופיעות במסמך. על פי כמות המילים המופיעות להגיד לאיזה מסמך שייך קובץ מסויים</p>

## תיאור החלופה הנבחרת

המערכת שנבחרה היא המערכת הלומדת. בחרנו במערכת הזאת בזכות העובדה שהיא מתאימה לדרישות שקבלנו מהמשרד. דרישות המשרד היו לבנות מערכת סיווג אמינה שתוכל להחליף את הבן אדם העובד במשרד.

סיבה ראשונה למה המערכת לומדת מתאימה. היא האמינות של המערכת, ניתן לראות מערכת לומדת הנמצאת בשימוש בתעשייה כדי לפתור בעיות מורכבות היא האמינה אפילו יותר מבן אדם. למשל המכונות האוטונומיות של טסלה היא בנויה מכונה לומדת. ולפי המחקר של המאמר <sup>6</sup> the byte, המכונות של טסלה נחשבת המכונות הכי בטוחה משאר המכונות הקיימות. וזאת בזכות העובדה שהיא משתמש במכונה לומדת.

סבינה נוספת לשימוש המערכת לומדת. היא שהחסרונות לא רלוונטים לדרישות של המשרד. הפרויקט זה הוא תהליך שעובד מאחורי הקלעים. היות כי מטרתו הוא לשמור על סדר במערכת. ולאחסן מידע ששומש כבר. לכן אין בעיה שלוקח למערכת המון זמן לעבוד. בנוסף לקח לחברה יש רק כמות מוגבלת של סוגי מסמכים. נדירים המקרים שבהם הם מוסיפים סוג חדש של מסמך. לכן אין צורך בלבנות מודל חדש למסמך נוסף. זאת הסיבה נוספת לשימוש במכונה לומדת.

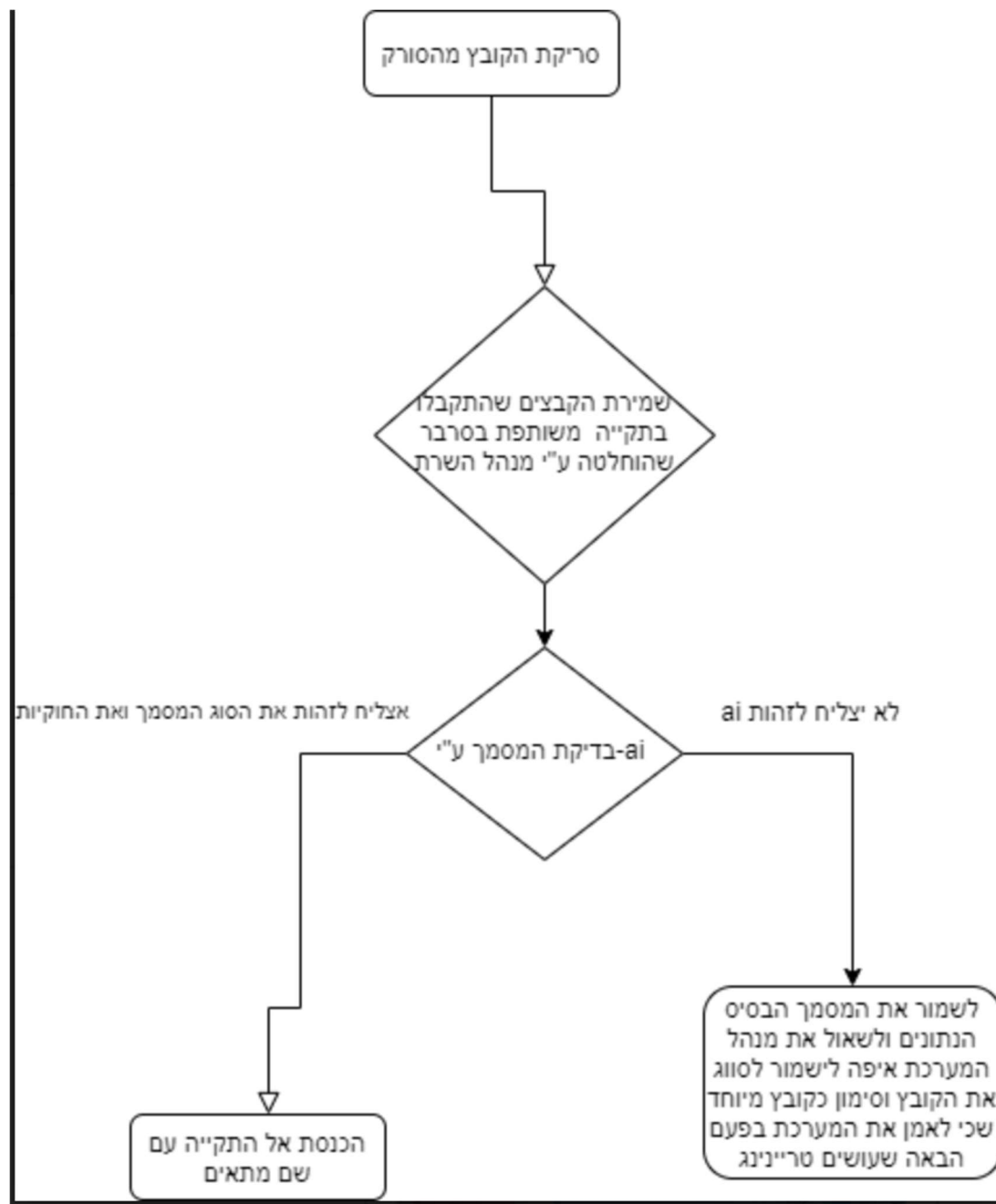
<sup>6</sup> [קישור למאמר של the byte](#)

## אפיון המערכת שהוגדרה

### ניתוח דרישות המערכת

- המערכת צריכה לקלוט מסמכים באזור המוגדר לה.
- הערכת צריכה לנתח כל מסמך ולהגיד לאן לסווג אותו.
- המערכת צריכה לשמור לאן כל מסמך עבר.
- המערכת צריכה ממשק משתמש לניהול והגדרות המערכת.  
ממשק שינוי מקום האחסון.
- ממשק לתיקיית המקום של הקבצים.
- ממשק לסיווג קבצים שהמערכת לא הצליחה לסווג.
- המערכת צריכה ממשק התחברות.

## מודל מערכת- תרשים זרימה לוגית של המערכת :



## אפיון פונקציונלי

### פירוק pdf למילים

כאשר אנו מקבלים את המידע אנו מקבלים אותו במסך pdf. לכן עלינו להוציא את כל המילים הנמצאים בתוך המסך. לכן עלינו לבדוק אם המסמך מכיל קובץ וורד תפרק את המילים שתבכו ושלח לתהליך הבאה. אחרת תבדוק אם הוא תמונה, ואם הוא תמונה אז תשתמש ב google ocr (שצויין בקרע הספרותי מה הוא עושה) ותוצאי את המילים שנמצאות בתמונה ותשלח אותו לתהליך הבא.

### ניקוי ותרגום המידע

כפי שציננתי אחד המין המרכיבים המרכזיים באימון מכונה. הוא שהמידע שהגיע למכונה יהיה נקי וללא שגיאות (כמו אותיות ספיות בתחילת המשפט או מילים ללא משמעות על זה הרחבתי גם אתגרים מרכזיים). בשלב זה אנו משתמשים בטכניקת ניקוי שגיאות הידועות mlp. בעזרת מודל של regular expression. אשר מוחק את המידע הלא רלוונטי כמו גם וסימנים כמו \$, & וכו'. בנוסף לכך אנו לא רוצים למחוק מידע שכתוב באנגלית. בזכות העובדה שישנם שמות חשובים שכתובים גם באנגלית ואנו צריכים להוסיף אותם גם. זאת ועוד את המספרים כמו תאריך או שנה או מספר סידרתי גם חשובים לנו. בשביל להוציא עוד מידע קריטי על המסמך. לכן נשתמש במודל תרגום מספרים למילים ונוסיף אותם לוקטור המילים.

### מיון מילים לפי שכיחותם

אחרי ניקוי המילים. אנו צריכים להכין את המודל שתוגש למוכנה. לכן על פי שיטת הכנתה מידע. אנו נמיינ את המילים לפי שכיחותם המופיעות במסך. את התהליך זה נעשה את זה בשביל אמצעי הinput של המוכנה. התהליך מתחיל בכך שהוא עובר מסמך מסמך. וכל מסמך הוא מדרג את המילים מי המילה השכיחה ביותר למידה הכי פחות שכיחה. אחרי שהוא עושה את פעולה זאת. הוא ממזג את כל הרשימות מילים לרשימה אחת ממוינת והיא תהיה template של יצירת מבנה הנתונים של כל המסמכים.

### מערכת לניהול תהליכים

אחת הבעיות הגדולות שהיו לנו בעת שלב הפיתוח הוא היה אמון המוכנה. והבעיה היא תהליך יצירת מסמך ללמידה היה לוקח זמן רב (פירות מלא פורט קודם בבעיה המרכזית). לכן יש לנו מערכת לניהול תהליכים ותהליכונים. הפונקציה זאת יוצר כמות תהליכים המחושבת על פי מספר הליבות שיש על המחשב שבו מותקן התוכנה. ולכל תהליך הגדרנו עוד 10 תהליכונים אשר מבצעים הכנת input למכונה הלומדת. דוגמא: אם במערכת יש לאמן 200 מסמכים. ויש למעבד 4 ליבות אז מספר הפעמים שהמליך הכנה לinput יתבצע הוא 40 בערך במקביל. לכן כל תהליך ותהליכון עושה את הפעולה רק חמש פעמים.

### יכולת חיזוי המסמך

אחרי שהמערכת הכינה לנו input ויש לנו מכונה לומדת אחרי שליו למידה. אז אנו מבצעים בדיקה על המסמך לפי ה input שניתן לו. בשלב זה אנו מפעילים את המכונה הלומדת אשר תיתן לנו תוצאה למה שייך המסמך. אחרי שנקבל תשובה מהמכונה אנחנו נבדוק מי קיבל את האחוזים הגבוהים ביותר

להתאמה. התאמה הגבוה ביותר הוא יהיה סוג מסמך שקיבלנו. ואת מה שקיבלנו אנו נשלח לשלב הבא כדי להוציא את הנתונים לשלב הבא.

## הוצאת הנתונים המסמך

היות אילוצי הזמן שלנו אז הנתון היחידי שנעבוד במסמך הוא שם הלקוח. בכל מסמך הוא מופיע בצורה שונה ובמקום אחר. אנו נשתמש בעוד מכונה של tensorflow לפי בנייתה של גוגל. שלמדנו אותה בעבר איפה נמצא שם הלקוח בכל מסמך והיא לבד תגלה בכל מסמך את שם הלקוח ותשלח את שמו ל list נתונים על הסמך.

## סידור מסמכים בתיקיות

המערכת לוקחת את הנתונים שהתכבלו מהניתוח של המכונה ולפיהם יוצרת תיקיות חדשות באיזור שנבחר ומסדרת בתוכם את הקבצים. קבצים בעייתיים לניתוח מועברים לתיקיה ספציפית, המשתמש יכול לבדוק בעזרת המערכת את ההקבצים האלה ולסדר אותם באופן ידני דרך הממשק הגרפי.

## לולאת אוטומציה

המערכת פועלת כלולאה שבא לאחר בדיקות DRM מתרחשת פעולת סידור המסמכים בצורה אוטומטית על התיקיה שנבחרה והקבצים מועברים בקבוצות למקומם המיועד עד שהתיקיה מתרוקנת. כשקבצים חדשים יופיעו בתיקיה הלולאה תחזור על עצמה ותסדר גם אותם. בזמן האוטומציה המערכת תציג למשתמש את הצב הנוכחי בשלבי הסידור.

## עידכון בסיס הנתונים

המערכת שומרת בבסיס נתונים מידע על כל לקוח וLOG על הקבצים שלו. המערכת יכולה להדכן ולשלות את המידע הזה ב real-time.

## התחברות והתנתקות

המערכת מחזיקה רשימה של לקוחות שכיבלו רישיון להשתמש בתוכנה ("קנו אותה"). בהפעלת התוכנה יש על הלקוח להתחבר למשתמש שלו ובסוף השימוש שלה תתרחש התנתקות אוטומטית מהמשתמש.

## הגדרות

הממשק הגרפי מספק חלון הגדרות בו ניתן לקבוע את התקיות הרלוונטיות לשימוש, רשימת קבצים שאין צורך להתייחס עליהם, לשפר את ההפניות של סידור הקבצים (אם לדוגמה לעובד בחברה יש שם כינוי או שם שה AI לא מצליח לנתח נכון) ואמצאי בדיקה של החיבור לרשת.

## ביצועים עיקריים

הביצוע העיקרי של המערכת הוא לסווג מסמכים שהוא מקבל במקום המתאים בשרת. המטרה של המערכת לעבוד ללא עומס האטה של השרת שהיא יושבת עליו. ונמצא כי

- רק בשלב הלמידה שהוא מחוץ לפעולת המערכת היה עומס קל על gpu
- בזמן ריצת המערכת אין עומס ופעולת סיווג המסמכים לוקחת בזמן סביר
- המערכת של הלקוח עובדת בצורה חלקה וברורה למשתמש ללא שום תקעות.

## אילוצים

### סביבת פיתוח:

- pycharm
- jupyter notebook
- eclipse

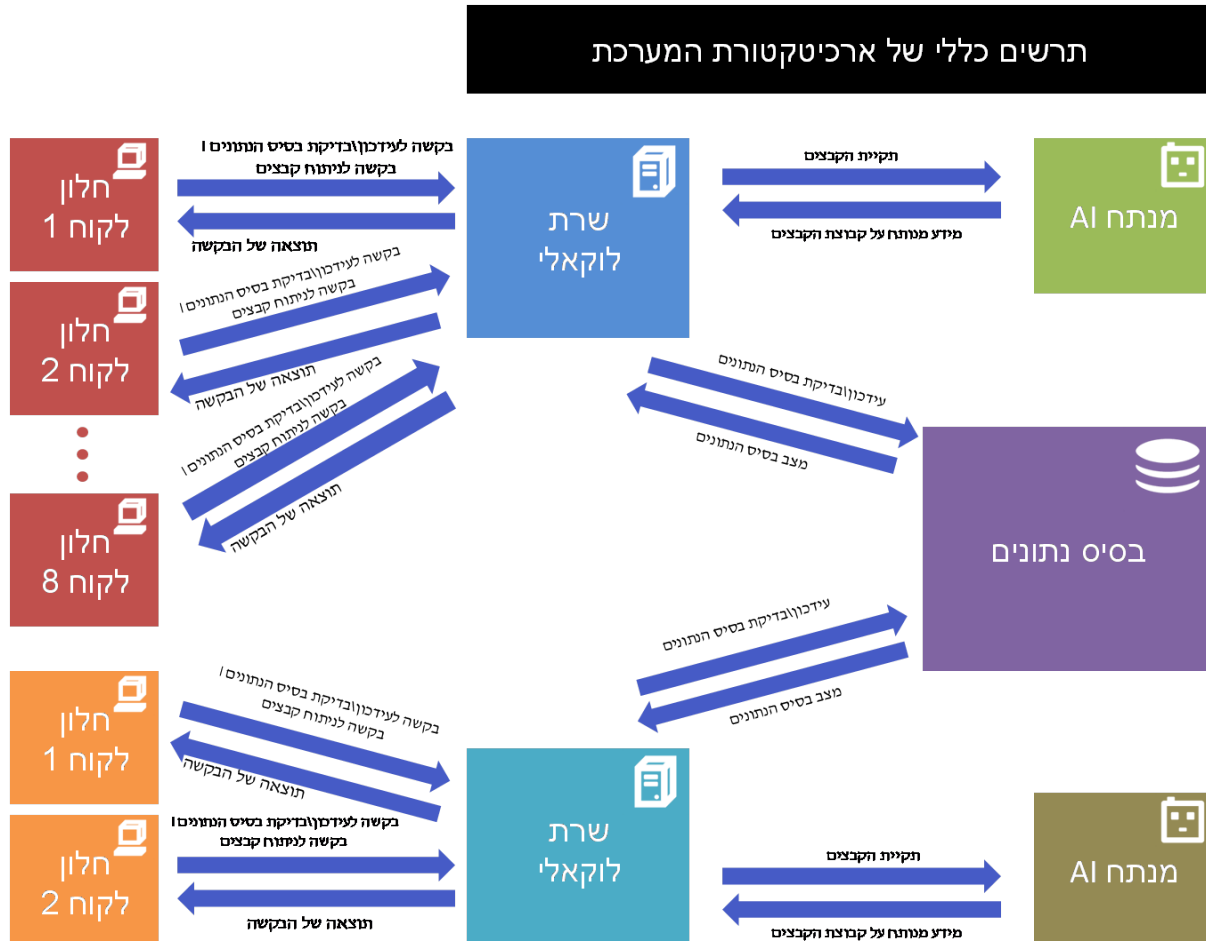
### כלי עזר:

- כלי לניהול דטה בייס-firebase
- כלי בנייה רשת נוירונים-tensorflow
- כלי להרצת קוד על מחשבים של גוגל-google colab
- oracle scene builder
- Adobe Photoshop

**(כל הגרפיקה בפרוייקט נעשת מ0 בפוטושופ ולא נלקח ממקורות חיצוניים על מנת שמירת האותנטיות של הפרוייקט)\***

# תיכון ארכיטקטורת המערכת

## תרשים כללי ארכיטקטורת המערכת



## תיאור הרכיבים בפתרון

- שרת לוקאלי-service שמגיע עם עותק של התוכנה ומודלק עם הפתיחה שלה. כתוב בPython ומשמש קשר בין הממשק לבסיס נתונים ובין הממשק למנתח קבצים.
- בסיס נתונים- בסיס נתונים NoSQL שפועל real-time על שרתים של Google בעזרת טכנולוגיית Firebase.
- לקוח/ממשק גרפי- ממשק גרפי שבנוי בעזרת טכנולוגיית JavaFX של תוכנת Oracle. הממשק מאפשר לבחור תיקיות, לסנן קבצים, לפתוח את הקבצים ולסדר אותם ידנית לפי רצון, כמו כן להמחיש בצורה גרפית את עבודת הAI.
- מנתח AI- קבוצה של מחלקות אשר עובדות על השרת ומעבות הלב של המערכת.

## תיאור התקשורת

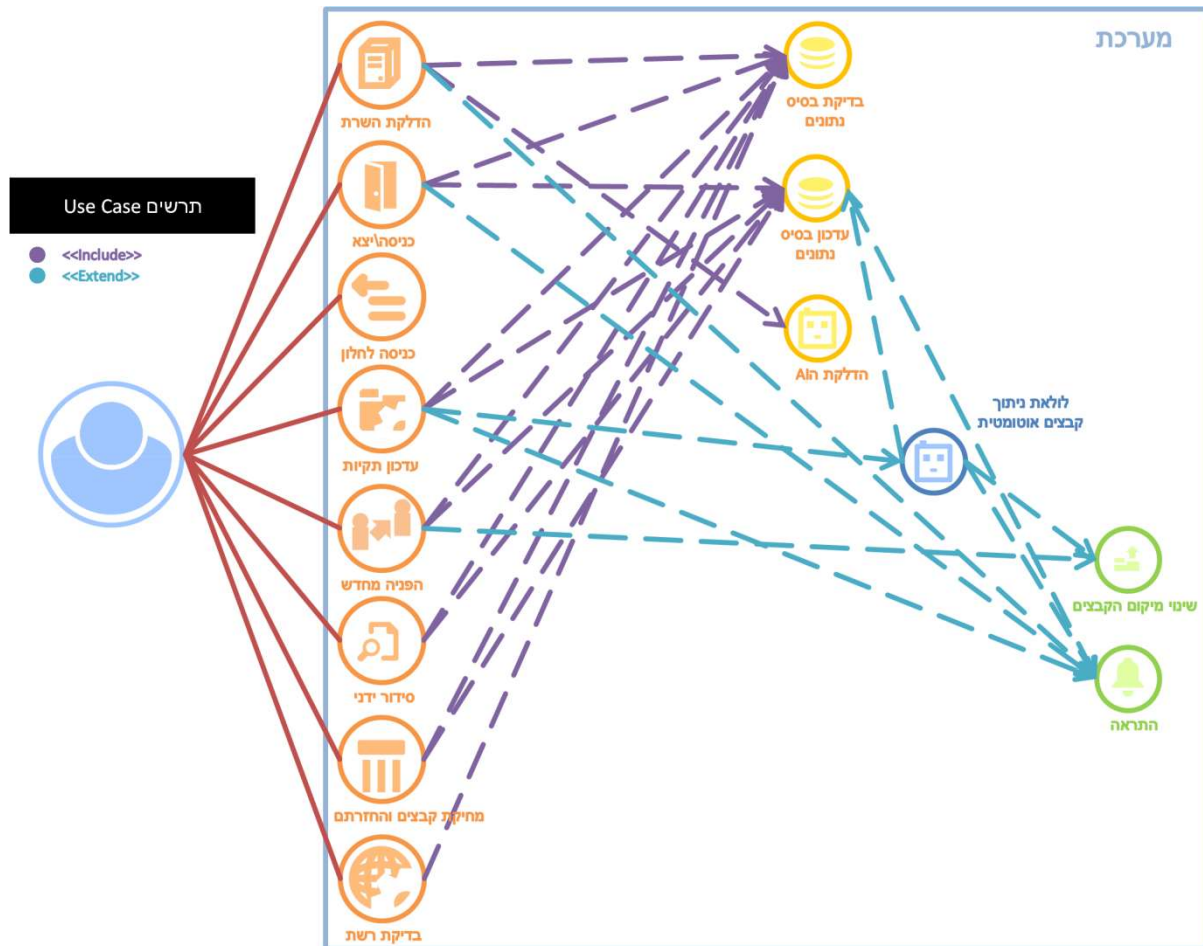
בתיקשורת בין הממשקים (החלונות הפתוחים) ושרתים הלוקים או השרת מרכזי של החברה מתבצעת בעזרת פרוטוקול TCP שמאפר להעביר כמות גדולה של מידע בצורה בטוחה ומהירה בלי שגיות או טעויות. בתקשורת בין השרת לבסיס הנתונים משומש הפרוטוקול WebSocket שמאפשר שיחה דו כיוונית מלאה (full-duplex) עם בסיסי הנתונים real-time של Google שפועלים על דפדן. הנתונים ברשת מעברים בין השרת והלקוח בשימוש בטכנולוגיית JSON.

השרת מהווה מקשר לא ישיר בין הבסיס נתונים והמנתח AI לממשק הגרפי ובכך גם מפצל את עבודת הממשק הגרפי והמנתח לתהליכים שונים ובכך משפר את האפקטיביות של המערכת. ניתן לפתוח מספר חלונות (ממשקים) במחשב אחד שיעבדו בו זמנית על מספר תיקיות וישתמשו במנתח AI יחיד מה שגורם ללמידה וכתוצאה לשיפור ביצועים של הAI עם הזמן כל עוד הוא פועל.

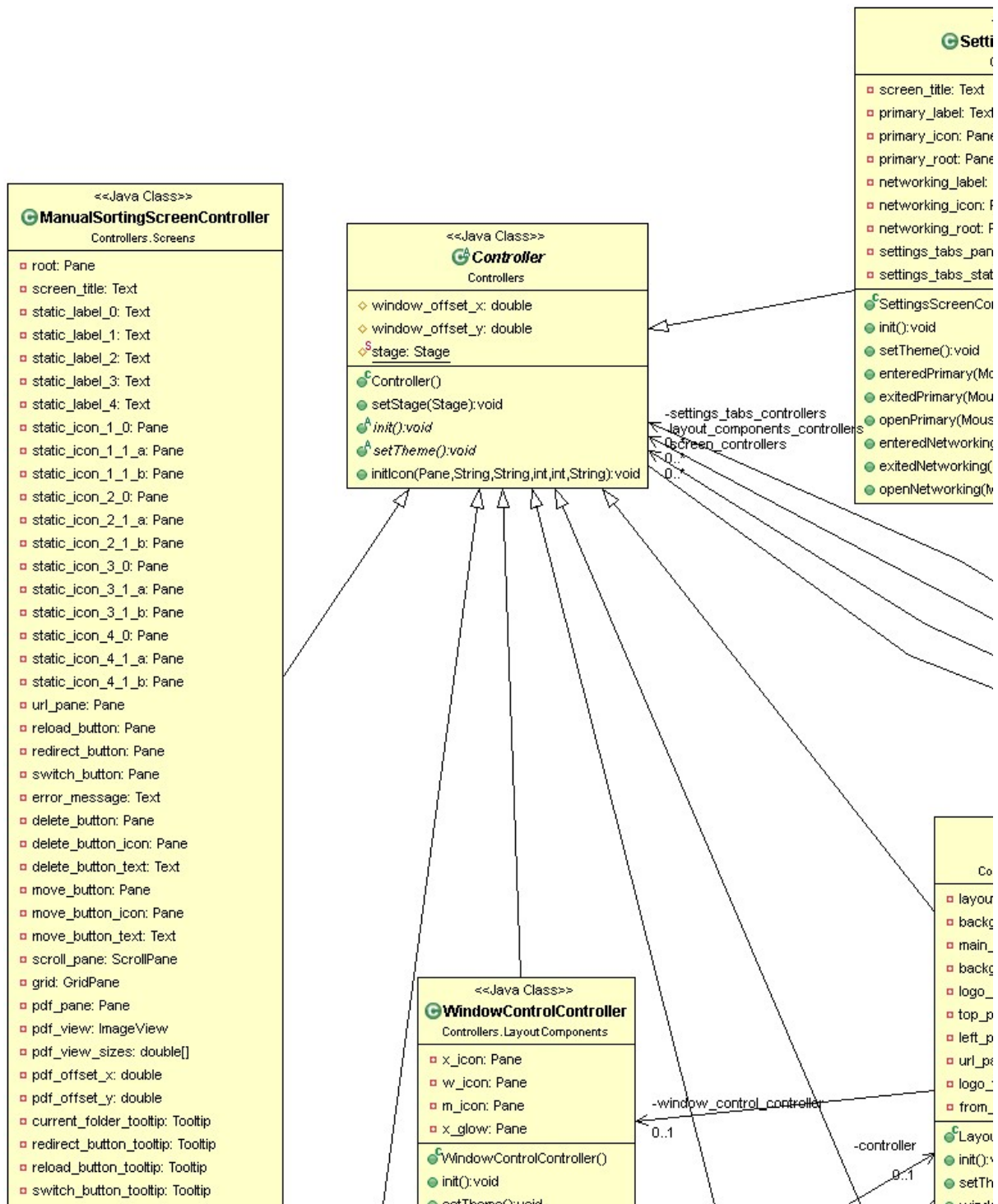


## ניתוח ותרשים Use / UML של המערכת המוצעת

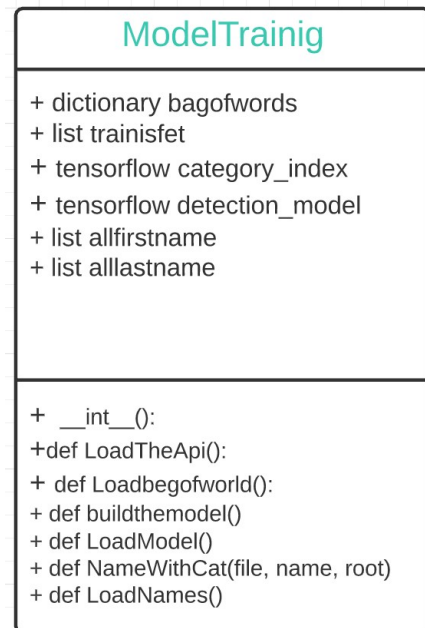
### תרשים Use Case



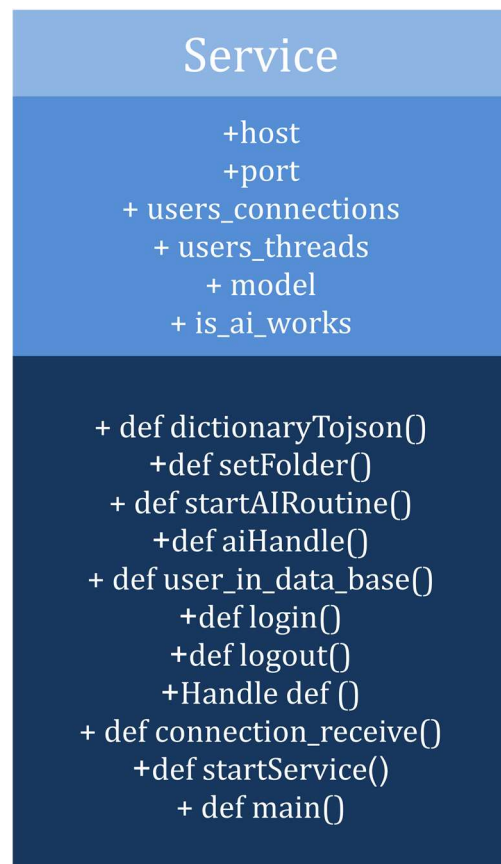
## תרשים UML של הממשק הגרפי



## תרשים UML של AI



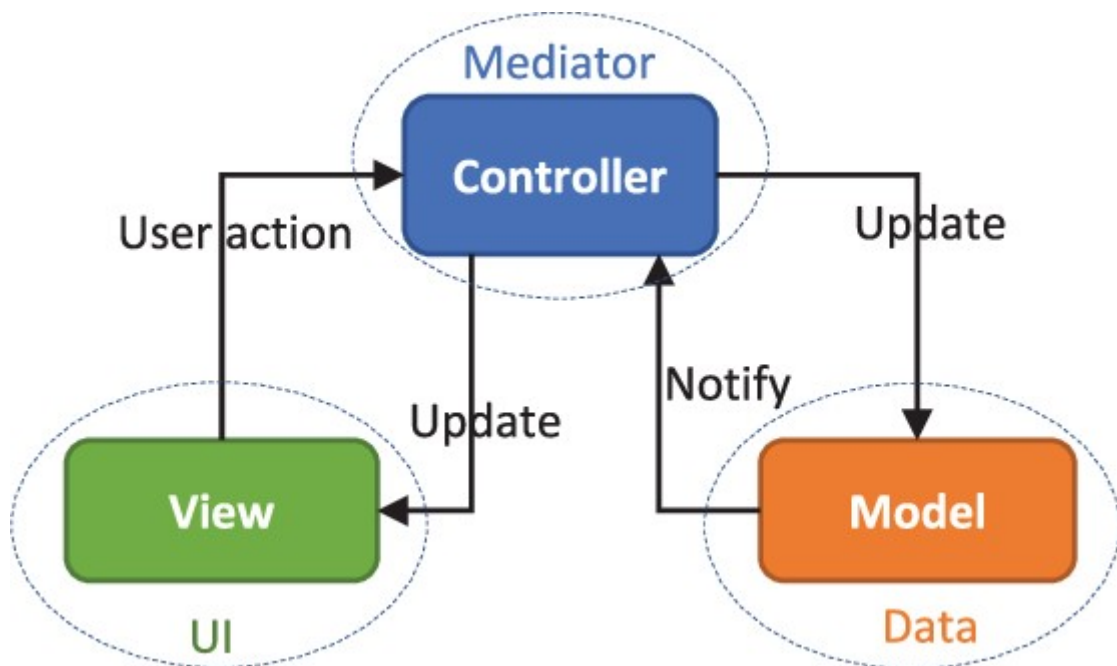
## תרשים UML של Service (שרת לוקלי)



## תיאור המחלקות המוצעות

### צד לקוח-ממשק גרפי

הממשק כתוב ב Java עם שימוש ב JavaFX על פי ה design pattern : MVC, משמה קיים פיצול בין החלק הוויזואלי לבין החלק החישובי/ מעשי אשר אותם מחבר שלט, חלק וויזואלי מפעיל בעזרת שלט את המודל.



לכן הצורה בה הממשק כתו היא כזאת- כל החלקים הוויזואלים כתובים ב FXML, לכל חלק כזה קיים מחלקות שלט והשלטים מחוברים למודל אשר בנוי בצורת סינגלטון.

- **Main** – מחלקה ראשית שמפעילה את האפקציה FX ופותחת את החלון המרכזי.
- **Controller** – אובייקט אבסטרקטי של שלט שעליו מתבססים כל השלטים.
- **LayoutController** – שלט החלון המרכזי.
- **MenuController** – שלט איזור התפריט בחלון המרכזי השוטת על בחירת המסכים.
- **ProgressController** – שלט סרגל ההתקדמות המציג וויזואלית את השלבים בהם נצא מנתח AI בשרת בזמן ביצוע ניתוח הקובץ.
- **ScreenRootController** – שלט על שורש המסכים בחלון המרכזי.
- **WindowControlController** – שלט על כפתורי שליטה על החלון המרכזי: כיבוי, הקטנה.
- **ManualSortingScreenController** – שלט על מסך הסידור הידני.
- **SettingsScreenController** – שלט על שורש מסך הגדרות התוכנה.
- **NotificationsScreenController** – שלט על מסך ההתרעות.
- **PrimaryTabController** – שלט על החלק המרכזי/ חלק הגדרת הקבצים ותיקיות במסך הגדרות התוכנית.
- **NetworkingTabController** – שלט על חלק התקשורת במסך ההגדרות של התוכנה.
- **PDFPreviewController** – שלט על חלק/ כפתורי ההצגה הממחשים את הקבצים בתיקיה במסך הסידור הידני.
- **LayoutComponents** – enum של החלקים המרכיבים את החלון המרכזי.
- **SettingsTabs** – enum של החלקים המרכיבים את מסך הגדרות התוכנה.
- **Screens** – enum של המסכים בתוכנה.
- **Model** – החלק המרכזי בצד הלקוח, מחובר לכל השלטים, אחרי על הלולאת התוכנית ובנוי ב design pattern : singleton.
- **ServiceConnector** – מחלקת הלקוח המתחברת לשרת הלוקלי בעזרת רשת בפרוטוקול TCP עם שימוש בJSON.
- **ThemeManager** – מחלקה המגדירה את הנושא הוויזואלי של הממשק הגרפי.

## צד שרת

חלק זה של המערכת כתוב כולו בשפת Python והוא החלק שעליו ממוקם מנתח AI, כמו כן זה הוא החלק המקשר את הלקוח לבסיס הנתונים.

- **Service** – מחלקת השרת (לוקלי) המתחברת ללקוח הלוקלי בעזרת רשת בפרוטוקול TCP עם שימוש בJSON ומחברת לבסיס הנתונים Firebase בעזרת firebase\_admin. מפילה את AI ומתפלת בלקוחות שלה במקביל בצורה של ריבוי-תהליכונים.
- **ModelTrainig** – מחלקה היוצרת ומטעינה/מאתחלת AI חדש שישימוש במערכת ויעבוד על השרת הלוקלי.

## מבנה נתונים:

פריוקט זה השתמשנו מספר רב של מבני נתונים בשביל משימות שונות. בזכות סוגי המני הנתונים האלה יכולנו להתאים לכל משימה את האלגוריתם המתאים לו.

## מילון:

הוא מבנה נתונים מופשט המגדיר אוסף של מפתחות וערכים. המילון מורכב ממיפוי חד-ערכי בין מפתח (Key) לערך (Value). הפעולה של מציאת הערך שמקושר למפתח מסוים נקראת חיפוש (ולעיתים גם שליפה), והיא הפעולה החשובה ביותר שמאפשר המילון. לדוגמה, ספר-טלפונים יכול להיות ממומש באמצעות מילון - מיפוי שמות של אנשים (מפתחות) אל מספרי הטלפון שלהם (ערכים). מילון בו המפתחות הם הערכים מגדיר קבוצה.

## שימוש במילון:

במילון השתמשנו כדי ליצר את המודל של bag of word. בזכות היכולות חיפוש של  $O(1)$  יכולנו להוסיף מילים שונים בקלות ולדעת עם יש מילה במערכת או איס מילה. ובוך הערך של המערכת ניתן לזהות כמה פעמים המילה הופיעה שם.

## מערך בולאני:

הוא אחד ממבני הנתונים הפשוטים ביותר: מערך הוא אוסף פריטים שניתן לגשת אליהם בצורה ישירה באמצעות אינדקס. האינדקס מסמל במערך זאת את המילה השכיחה ביותר שנימצאת במליון וסמנים ב-1 אם היא קיימת או ב-0 אם היא לא קיימת

## מדריך למשתמש-תיאור ה-GUI (מסכים וכפתורים)

### Login

בעת הפעלת ההמערכת תכילה המשתמש מוצג למסך הכניסה :

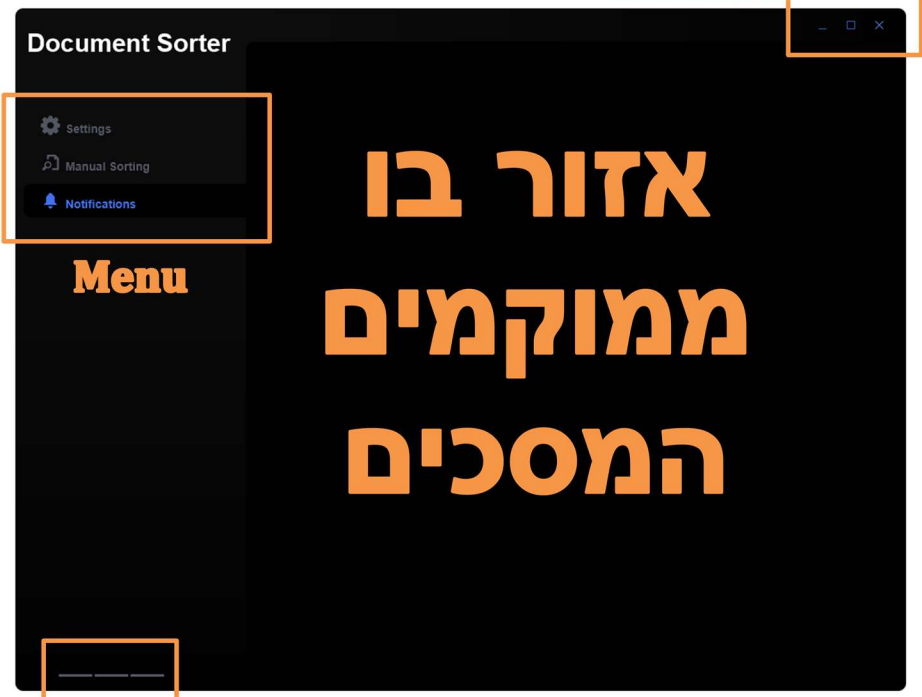


הבוי משתי תיבות הכנסת טקסט וכפתור הכניסה.  
לאחר הכנסת האישורים שלו לתוך תיבות הטקסט כפתור הכניסה ידלק :



וכעת הלחיצה עליו תעדכן את מצב המשתמש בבסיס הנתונים ותאתחל את הנתונים שלו מהזיכרון (אם הם קיימים)

### החלון המרכזי



**כפתור ה Exit minimize**

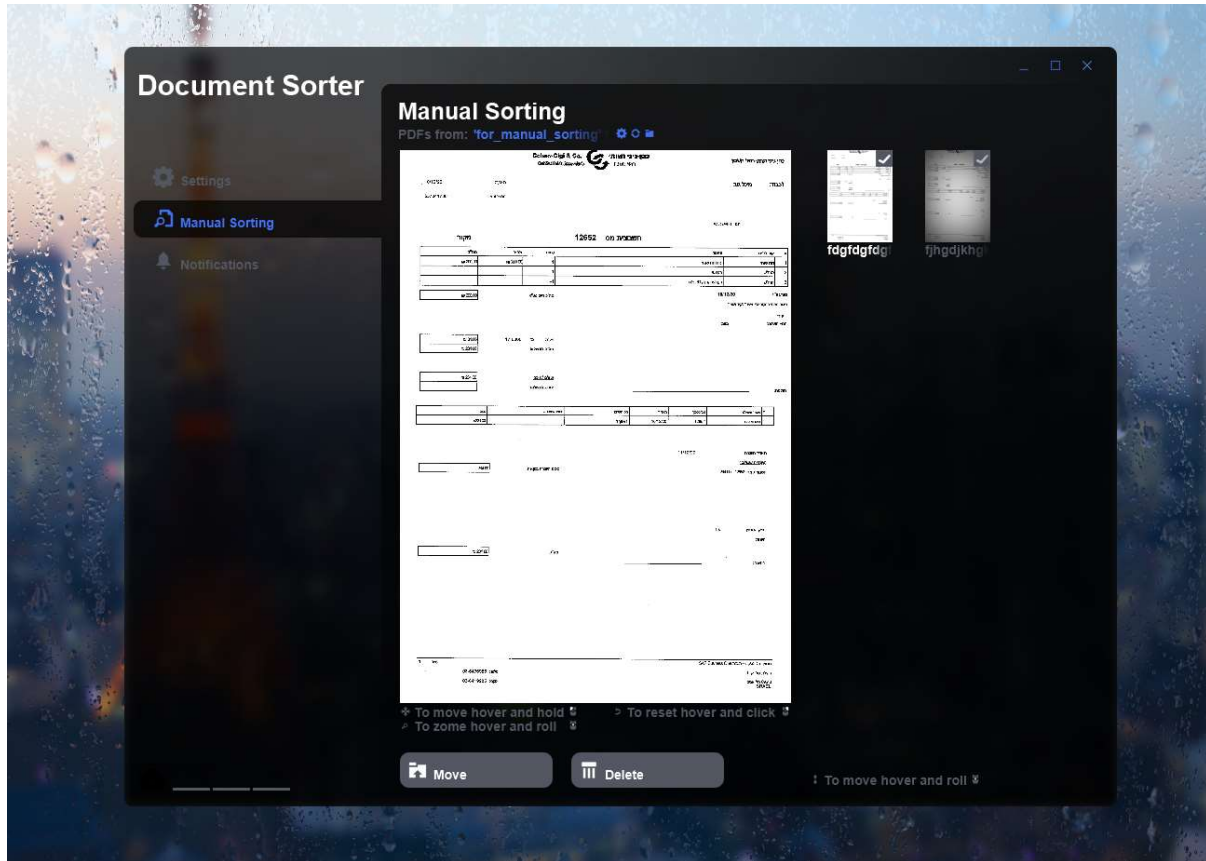
**Menu**

**אזור בו ממוקמים המסכים**

**מדי המתאר את המצב הניתן האוטומטי (סרגל סעינה)**

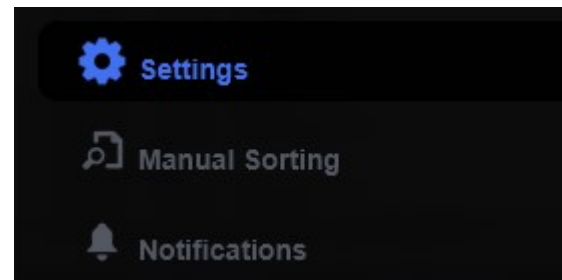


בעזרת פונקצייה המצלמת את המסך לא אחר הזזת החלון עליו, קיימת לממשק גם האפצייה לTheme שקוף המחקה זכוכית (השראה הגיע מהAero Glass של Microsoft)








## Menu

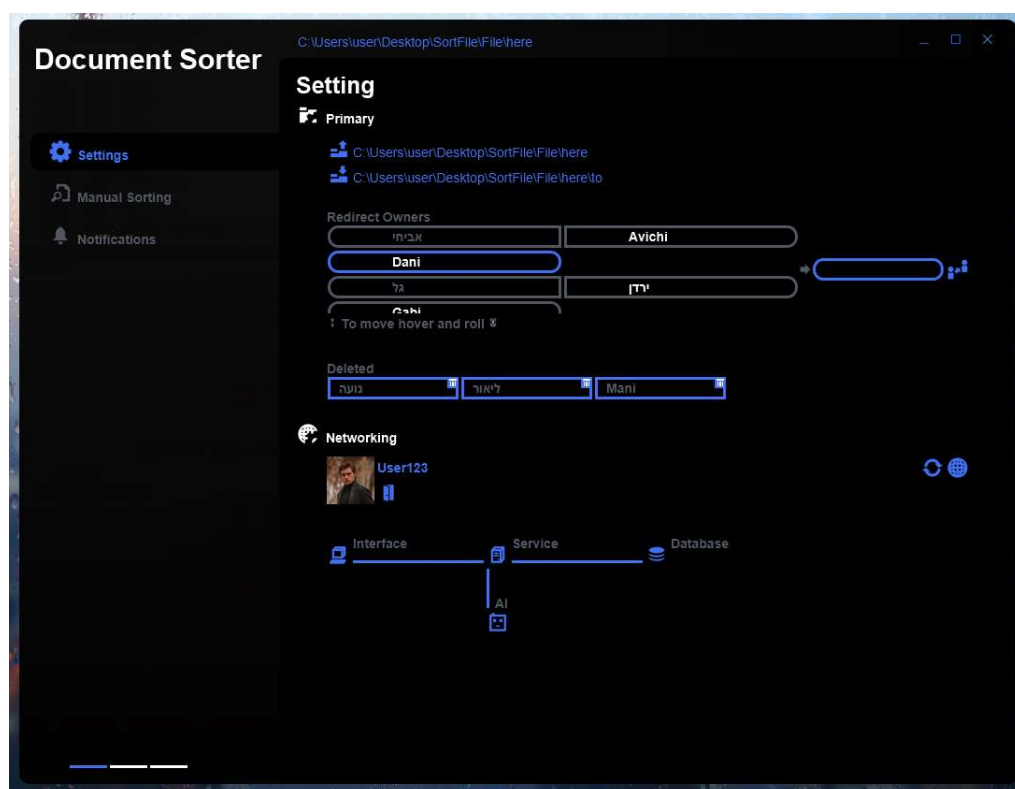


תפריט מרכזית המורכבת מ3 כפתורים ומאפשרת לעבר בין המסכים השונים :


- Settings – הגדרות 
- Manual Sorting – סידור קבצים ידני 
- Notifications – התראות 

## Settings

המסך הראשון והכי מרכזי זה הוא מסך ההגדרות. במסך המשתמש קובע את התיקיות בהם רוצה להשתמש, מפרט את הקבצים ואת הדרך שעליהם להיות מסודרים. כמו כן בודק את החיבור שלו לרשת ומאתחל אותה אם יש צורך. כדי לפתוח כל אחד מהחלקים של המסך ההגדרות יש לחוץ על הכפתורים שמסמנים את החלקים האלה.




## חלק ראשון - Primary

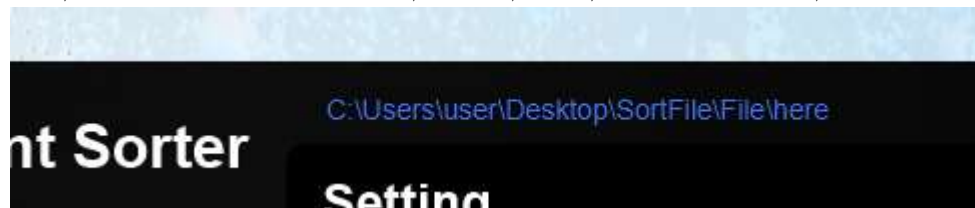
חלק זה מסומן על ידי  וכמו שניתן להבין מהשם שלו הוא החלק המרכזי של ההגדרות, וכמו שניתן להבין מהמסל שלו אלה הם ההגדרות שקשורות לתיקיות וקבצים המעברים.




חלק זה מחולק לשלוש:

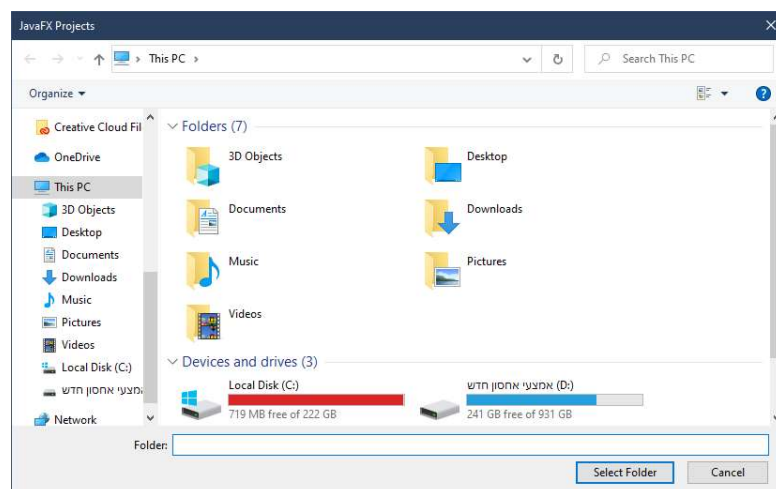
❖ הראשון הוא בחירת תקיית היציא ותקיית הכניסה ומוכב משני כפתורים.

- כפתור בחירת תקיית היציא  תקייה זאת תשמש מקום מאיפה המערכת תקח קבצים ותסדר אותם. לאחר בחירת התיקיה המארכת אוטומטית תיצור בתוכה תיקיה שתשמש מיקום לכל הקבצים הבעייתיים שהAI לא הצליח לסדר ותעדכן את מסך הסידור הידני עם הקבצים בתוך. כמו כן למשם נוכות בחלק העליון של חלון הGUI יופיע מסלול מלא לתקייה זאת.



- כפתור בחירת תקיית הכניסה  תקייה זאת תשמש מקום/שורש איפה שהמערכת תצור תיקיות ותסדר לתוכן את הקבצים המסודרים אוטומטית.

בחרת התיקיות תתבצעה על ידי חלון בחירה מקומי למערכת ההפעלה אשר פשוט ומובן לכל המשתמשים של מערכת הפעלה הזאת.



במקרה והשימוש במערכת הוא לא ראשוני, לאחר כניסה המערכת אוטומטית תעדכן את התיקיות האלה לאלה ששימשו בהפעלה האחרונה מבסיס הנתונים.

במקרה והתרחשה שגיאה והתקיימה כבר לא קיימת על המחשב המערכת תודעה על כך למשתמש.


❖ **Redirect** או הפניה מחדש, היא אפצייה בהגדרות שמתרת לשפר את הביצועים של הסידור האוטומטי. אם בשביל לתקן שגויות בשמות שה AI מתקשה לקרוא או לסדר קבצים של קבוצת אנשים למקום אחיד, כלי ההפניה מחדש יכול מעוד לשמש את המשתמש בתוכנה. הוא בנוי מרשימה של כל ההפניות וכל הבעלים שהקבצים שלהם עברו בדיקה על ידי AI, תיבת

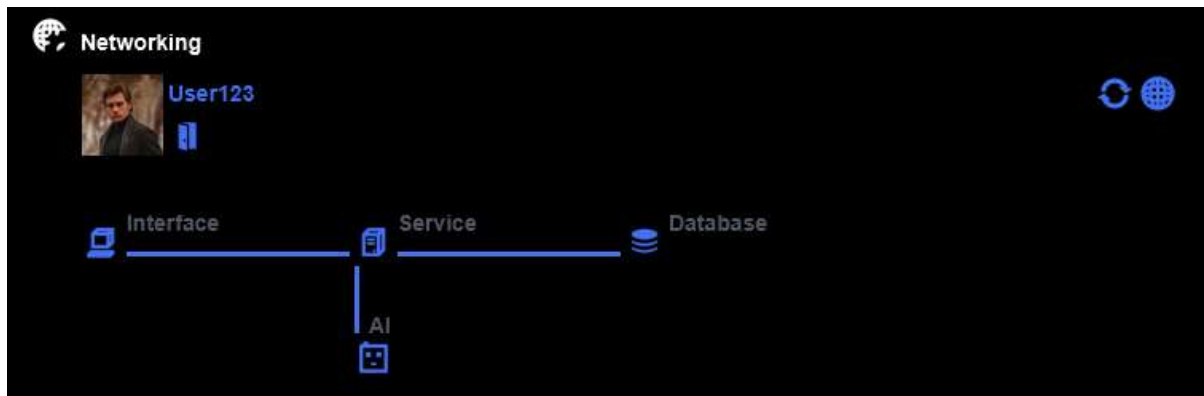
טקסט שבא כותבים את ההפניה החדשה וכפתור  שלאחר לחיצה עליו מבצא את העדכון. מעבר לעדכון במבנה הנתונים לשם המיקוד העתידי של התוכנה, המערכת גם אוטומטית תעדכן את כל הקבצים שסודרו בעבר למקומם הנכון החדש.

❖ ואחרון חביב סרגל הקבצים המחקרים. קבצים אלה אינם מחוקים מהזכרון של המחשב לשם לא לגרום לעיבוד קבצים שיכול לפגוע במשתמש אך המערכת תבחר לא להתייחס לקבצים אלה ולתציג אותם או תנסה לסדר אותם למקום חדש. אפשרות המחיקה מופעה במסך אחר, סידור ידני, אך בעזרת סרגל זה ניתן לאחזיר קבצים שנמחקו בטעות למרחב הראה של המערכת. כל מה

שעל המשתמש לעשות זה ללחוץ על כפתור ה  ליד כל השם של קובץ והמערכת תעדכן אותו עצלה.


## חלק שני- Networking

חלק זה מסומן על ידי  וכמו שניתן להבין מהשם שלו הוא החלק בהגדרות המתעסק בניעול הרשתות.








חלק זה גם הוא מחולק לשלוש:

❖ הראשון הוא פינה שמפשרת לראות איזה משתמש מחובר עכשיו למערכת ומשתמש בתוכנה.

מורכב מתמונה של המשתמש, שם וכפתור בצורת דלת  שבלחיצה עליו המשתמש מתנתק מהתוכנה (\* חשוב לדעת כי ללא חיבור למערכת התוכנה תסרב לעבוד עד לכניסה חדשה מהמשתמש או לקוח אחר).

❖ חלון הממשל הצגה וויזואלים של החיברים והחלקים הפעילים ברשת:

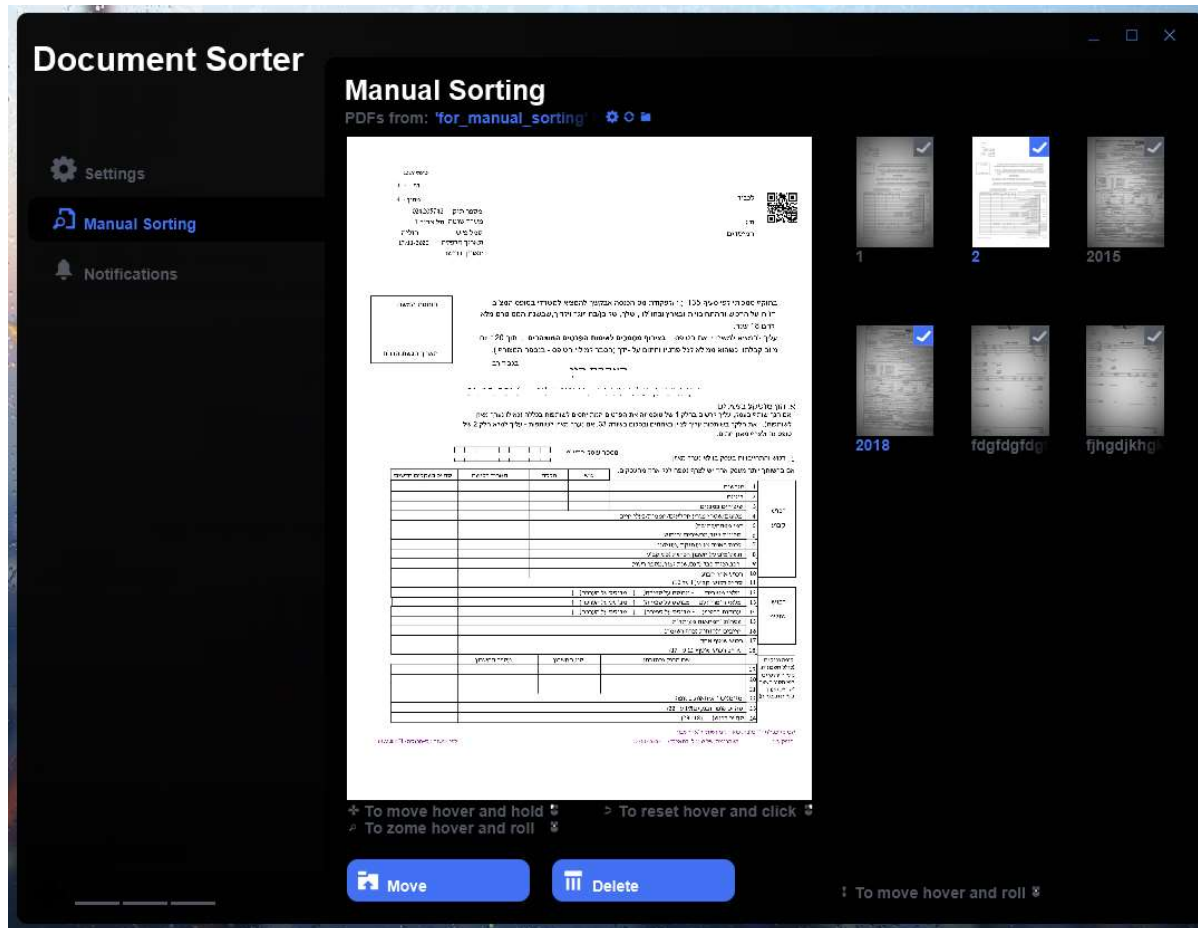
- הלקוח-  ממשק גרפי שכמובן תמיד פועל.
- השרת הלקלי המסומל על ידי  דלוק כאשר קיים עליו חיבור.
- AI מנתח אוטומטי שפועל על השרת ומסומל על ידי  דלוק כאשר סיים להתאחל על השרת.
- ובסיס הנתונים המסומל על ידי  דלוק כאשר קיים עליו חיבור.
- ❖ פינה שמכילה שתי כפורים:

- כפתור לבדיקה וויזואלית מחודשת של כל החיבורים. 
- כפתור להדלקה חדשה של השרת הלקלי במקרה והוא נסגר בטעות על ידי המשתמש.






## Manual Sorting

לאיתים רחוקות קיימים מקרים בהם ה-AI לא מצליח לנתח עד הסוף את הקוץ, או שהמשתמש מחליט יש קבצים ספציפים שלא צריכים לעבור את הסידור האוטומטי, לשם כך גם קיים במערכת מסך לסידור ידני. המסך בנוסף מאפשר לבדוק ולקרוא את הקבצים שעדיין לא נותחו או קשלו בלי לפתוח אתם בחוץ תוכנת PDF חיצונית.



### חלק ראשון:

- המסך מכיל תווית שמציגה איזה תקייה מפותת ברגע זה
- כפתור  אשר מעביר מהר את המשתמש להגדרות וישר לשינוי תקיית היציא
- כפתור  שמעדכן את הקבצים בתקייה שמוצגים על המסך
- כפתור  שמחליף על המסך בין הקבצים בתקיית היציא והקבצים שלא הצליחו לעבור ניתוח מושלם על ידי ה-AI, והאפק.

## חלק שני:

המסך מחיל חלון אשר מתרגם את את קבצי הPDF לתמונות ומציג אותם למשתמש בשביל בדיקה וקריאה.

- בעזרת כפתור שמאלי של העכבר ניתן להזיז את הקובץ המוצג
- בעזרת כפתור מרכזי של העכבר אפשר להתקרב ולהתרחק
- בעזרת כפתור ימני של העכבר למצב הראשוני ברגע אחד

## חלק שלישי:

כל הקבצים בתיקיה המוצגת יופיעו בסרגל רשת שניתן לגלגל בעזרת העכבר.



- הקבצים יופיע ברשת בצורה הבאה (משמאל לא מסומנים ולא מוצגים על המסך לקריאה).





- ניתן לסמן אותם על ידי לחיצה על הריבוע בצד הימני עליון שלהם והשם נדלקים ניתן להבין שהקובץ סומן.



- כדי להדליק על מסך בשביל בדיקה/ קריאה יש צורך פשוט ללחוץ על הקובץ הקובץ המוצג על המסך יהיה בלי החשכה והשם יהיה דלוק.

## חלק רביעי:

אם יש לפחות קובץ אחד מסומן הכפתורים בתחתית המסך ידלקו ונתן יהיה ללחוץ עליהם.

-  כפתור ה שאינו מוחק את כל הקבצים המסומנים אך גורם למערכת להתעלם מהם, כדי שוב לחזור ולהשתמש בהם יש להכנס להגדרות המרגזיות להחזיר אותם שם.
-  כפתור ה פותח חלון בחירת תיקיות של מערכת הפעלה ובעזרתו מעביר את כל הקבצים המסומנים למיקום החדש שנבחר.

## Notifications



כשהסמל של המסך הזה מכבל נקודה אדומה יש להכנס למסך הזה ושם תופיע התרעה על בעיה או על חלק חסר שהמשתמש צריך לדעת וקישור לתקן אותו (תיקיה לא מוגדרת או קבצים שיש לסדר באופן ידני)

## ניתוח קוד התוכנית

### ספר 1: יצירת וקטור מילים

pdf בספר זה נדבר על תהליך יצירת המילים. אנו נראה איך הגענו מקובץ

יש באו תמונה. לוקטור בעל מילים משמעותיות בשביל ליצור אינפות למכונה כמו שדברנו בספר הראשון

הספר זה מוכרב שני קבצי פייתון שהופכים לספר אחד

#### import

In [44]:

```
import os #ספרייה שנותנת לנו לעשות פעולות של מערכת הפעלה
from pdf2image import convert_from_path #לתמונה pdf ספרייה שהופעת לנו
import pytesseract #של גוגל מוצאיה מילים מתמונה ocr ספרייה
import CleanText #קובץ שני של ספר 1 ניתן להתעלם ממנו בשלב זה
from collections import Counter #פעולות לאיחוד בני נתונים מורכבים
from numba import jit #ספרייה שנות לנו לקופילציה למעבד
import re #ספרייה regular expression
from num2words import num2words #תרגום מילולי של מספרים
#from googletrans import Translator
from google_trans_new import google_translator #ספריית תרגום של שפות העולם
import HebrewNlp #קובץ נוסף לספר 1
import config #דף הגדרות מערכת
```

#### הוצאת מילים מתמונה

הפונקציה זאת מקבלת מיקום של תמונה.

הפונקציה מחזירה את המילים המופעים בתוך התמונה

In [45]:

```
@jit(parallel=True) #הוראת מכונה חפעת קוד על ה-gpu
def imageToword(PicName):
    pytesseract.pytesseract.tesseract_cmd = config.PYTESERACT #פעולת הפעלה
    ocr של גוגל
    return pytesseract.image_to_string(os.path.join(config.WORKSPACE,
    PicName), lang=config.PYTESERACT_LANGUAGE)
```



## pdf הוצאת תמונה מ

הפעולה מקבלת כתובת של הקובץ, שם קובץ, משתני בולייני אם התמונה נועדה לחתוך

הפעולה מחזירה מיקום של קובץ של תמונה או את המלים שבתוכו טלוי במשתנה הבוליאני

In [46]:

```
def ReadPdf(root,name,iscrop):  
    pages = []  
    pages = convert_from_path(root,  
    poppler_path=os.path.join(config.POPPLER))  
    (Name,file) = name.split('.')  
    PicName =Name+"."+config.IMAGETYPE  
    pages[0].save(os.path.join(config.WORKSPACE, PicName),  
    config.IMAGETYPE)  
    if iscrop:  
        return os.path.join(config.WORKSPACE, PicName) # המטרה של החלק הזה  
        היא בשביל חלק אחר של הפרויקט נדבר על זה בהמשך  
    else:  
        words = imageToword(PicName)  
        os.remove(os.path.join(config.WORKSPACE, PicName)) # אחרי קבלת  
        המילים נמחק את התמונה בשביל לא לתפוס מקום  
        return words
```

## ניקוי המילים

אחרי שהתאנו את המילים המקובץ אנחנו רוצים להפטר מצרופים לא הגיונים אשר מורידים את יעילות המכונה

In [47]:

```
# הפעולה מקבלת טקסט גולמי
# הפעולה מחזירה את הטקסט נקי
def CleanTheText(Text):
    Text = " ".join(Text.split())
    Text = Translate2Word(Text)
    Text = CleanRound1(Text)
    return Text

# הפעולה מקבלת טקסט עם מספרים ואתיות באנגלית
# המכונה מתרגמת את המספרים והאותיות באנגלית לשפה בנחרת
def Translate2Word(text):
    # translator =
    Translator(['translate.googleapis.com', 'translate.google.com', 'translate.google.co.kr'])
    translator = google_translator()
    wordsarr = text.split(' ')
    for i, element in enumerate(wordsarr): # לולאה אשר בודקת את כל המילים בטקסט
        if(element.isnumeric()): # בודקת שזה מספר ותרגמת אותו
            text2 = num2words(int(element), lang='en')
            #text = translator.translate(text2, dest='he').text
            text =
    translator.translate(text2, lang_tgt=config.SYSTEMLANGUAGE)
    wordsarr[i] = text
    text = " ".join(wordsarr)
    return text

# הפעולה מקבלת טקסט גולמי
# הפעולה מחזירה טקסט נקי ללא לסמנים מיוחדים או רווחים מיותרים
def CleanRound1(Text):
    Text = re.sub('[...]', '', Text) # נקיו סוגרים
    Text = re.sub('[%s]' % re.escape(r'!#$%&*~\./,©-() :;<=>[]^_--`?{|}~."\''), '', Text) # נקוי סמנים מיוחדים
    Text = ' '.join([w for w in Text.split() if len(w)>1])
    Text = re.sub('[\d\p{QAM}^]', '', Text) # מחיקת מלים אשר מתחילות או באמצע המילה יש אותיות ספיות
    Text = re.sub(' ', '', Text)
    Text = re.sub(' ', '', Text)
    Text = re.sub('\d+', '', Text) # מחיקת אנטרים
    return Text
```

## Tokenizer

הפקונרציה זאת לוקחת את כל המילים במערך ומשנה את המילים למילים של השורש למשל מהנדסים מהנדסות הופכים ל מנהדס

In [48]:

```
def HebrewTokenizer(Text):
    try:
        request = {"token": config.TOKENHEBNLP, "text": Text} # לקריאה ל api
        # חיצוני שיעשה את זה
        return requests.post(config.TOKENIZERURL, json=request).json()[0]
    except:
        array = Text.split(' ')
        return array
```

## שימוש כל הפונקציות

הפעולה מקבלת כתובת של קובץ ושם קובץ

הפעולה מחזירה וקטור של מידע נקי של מילים.

In [49]:

```
def Tokenize(root, name):
    fileName = ReadPdf(root, name, False)
    text = CleanText.CleanTheText(fileName)

    tokenizer = HebrewNlp.HebrewTokenizer(text)
    for i, element in enumerate(tokenizer): # מאלף יותר מאות
        # אחרת
        if (len(element) <= 1):
            tokenizer.pop(i)
    return tokenizer
```

## פקנציות נספוח שהספר הזה יכול לעשות

In [50]:

# הפעולה מקבלת ווקטור מיליון, מערך של כל שמות הפרטים, מערך של כל שמות המישפחה  
# הפונקציה מחזירה שם מלא בכתוב בווקטור

```
def GetName(text, allname, allfname):
    if type(text) == str:
        array = text.split(" ")
    else:
        array = text
    i = 0
    firstname = ""
    lastname = ""
    havefullname = False
    while i < (len(array)) and not havefullname: ## חפושם הם חפושם
        if lastname == "" and not (i+1 == len(array)):
            two = array[i] + " " + array[i+1]
            if (two in allfname):
                lastname = two
                i += 1
            elif str(array[i]) in allfname:
                lastname = array[i]
            elif (firstname == "" and i+2 < len(array)):
                two = array[i+1] + " " + array[i+2]
                if (str(array[i]) in allfname and two != "שם האב"):
                    firstname = array[i]
            if (not (firstname == "") and not (lastname == "")):
                havefullname = True
            i += 1
    if havefullname:
        return firstname + " " + lastname
    else:
        return config.ERRORLABEL
```

# הפונקציה זאת מקבלת תמיקום של תמונה מערך של כל שמות המישפחה ומחזירה את השם המלא של אדם

```
def GetName(PicName, allname, allfname):
    text = imageToword(PicName)
    new_text = " ".join(text.split())
    return HebrewNlp.GetName(new_text, allname, allfname)
```

# הפעולה מקבלת שני מילונים

# הפעולה מאחדת את שני המילונים השונים

```
def margedictionaries(dictionary1, dictionary2):
    ini_dictionary1 = Counter(dictionary1)
    ini_dictionary2 = Counter(dictionary2)
    return ini_dictionary1 + ini_dictionary2
```

## בניית הדטה בשביל אימון המוכנה

אחרי שהצלחנו לזהות את המילים ונקינו אותם אנחנו רוצים לבנות רשימה של כל הנתונים החשובים

לכן יצרנו פונקציה אשר בונה את המידע החשוב

In [51]:

```
def BuildModelToTraining(root,name):
    #פונקציה מקבלת שם קובץ ומיקום קובץ
    #פונקציה מחזירה רשימה של סוג מסמך, מיקומו, והמילים שיש בפנים
    #מחזירה מילון שיכולות מילים
    rap = ()
    #try:
    if 1==1:
        wordfreq = {}
        wordtoken = Tokenize(root,name)
        for token in wordtoken: # לולא בונא את מילון שכותבת כמה הופעות יש
            #לכל מילה
            if token not in wordfreq.keys():
                wordfreq[token] = 1
            else:
                wordfreq[token] += 1
            rap = [config.ERRORLABEL, root, wordtoken]
            if config.REPORTTYPE1 in root: # באימון המכונה אז בכתוב
                #היה כתוב איפה ניצא הקובץ
                rap = [config.REPORTTYPE1, root,wordtoken]
            if config.REPORTTYPE2 in root: # באימון המכונה אז בכתוב היה
                #כתוב איפה ניצא הקובץ
                rap = [config.REPORTTYPE2,root, wordtoken] # לקובץ חדש זה יכתוב
            #לבדיקה
        return [rap,wordfreq]
    # except Exception as e:
    #     return [config.ERRORLABEL,root,config.ERRORLABEL]
```

## תוצאה של כל התהליכים יחד

הפעולה יכולה לקחת כמה דקות.

In [52]:

```
import warnings
warnings.filterwarnings("ignore")
print(BuildModelToTraing(os.path.join("exmple", "2015.pdf"), "2015.pdf"))
טופס, 'אל', 'שלוש', 'מאות', 'ואחד', 'י', 'exmple\\2015.pdf', ['לבדיקה', 'י', 'שודר', 'עי', 'מייצג', 'באינטראנט', 'אחד', 'מתו', 'ארבעה', 'התשמיס', 'ליחיד', 'ווי', 'ווי', 'אסמכתא', 'חותמת', 'המשרד', 'הרשו', 'בלשו', 'זכר', 'מתיחס', 'לנקבה', 'די', 'וחשבו', 'על', 'ההכנסות', 'באר', 'ובחול', 'תשע', 'תשע', 'תארי', 'הגשת', 'הדוח', 'ור', 'הדוח', 'בשנת', 'המס', 'אלפיי', 'ושמונהעשרה', 'שתיי', 'השנה', 'המתחילה', 'והמסתיימת', 'השומה', 'תל', 'אביב', 'חמש', 'חוליה', 'שבעה', 'עפי', 'סעי', 'מאה', 'ושלושי', 'ואחת', 'לפקודת', 'מס', 'הכנסה', 'רשאי', 'פקיד', 'השומה', 'לראות', 'מי', 'שהגיש', 'דוח', 'שלא', 'כלליי', 'מולא', 'כראוי', 'או', 'שלא', 'צורפו', 'אליו', 'המסמכי', 'המתאימי', 'כמי', 'שלא', 'הגיש', 'דוח', 'על', 'הכנסות', 'והכנסות', 'בת', 'זוגי', 'שלושה', 'הכנסות', 'בלבד', 'חמש', 'אני', 'מגיש', 'דוח', 'לשנת', 'מס', 'זו', 'למרות', 'שאיני', 'חייבבקשה', 'להחזיר', 'מס', 'וגי', 'מגיש', 'דוח', 'נפרד', 'מצב', 'הדוחהצהרה', 'של', 'בת', 'זוגי', 'אי', 'הכנסה', 'לב', 'בת', 'זוגי', 'בת', 'זוגי', 'עזר', 'לי', 'בהשגת', 'הכנסה', 'תארי', 'הגעה', 'זוג', 'רשו', 'סה', 'משות', 'לבני', 'הזוג', 'ללא', 'כי', 'עמדתי', 'בתנאי', 'סעי', 'לפקודה', 'שלושה', 'לא', 'עמדתי', 'בתנאי', 'סעי', 'לפקודה', 'עולה', 'חדש', 'תושב', 'חוזר', 'ותיק', 'רתושב', 'חוזר', 'וחלות', 'על', 'הכנסות', 'מחול', 'הקלות', 'במס', 'ההדהדק', 'לקדה', 'קמס', 'היו', 'לי', 'או', 'לב', 'בת', 'זוגי', 'או', 'לילדי', 'שטר', 'מלאו', 'לה', 'נכסי', 'בחול', 'בשווי', 'של', 'שח', 'או', 'יותר', 'המס', 'היתה', 'לי', 'או', 'לב', 'בת', 'זוגי', 'הכנסה', 'חייבת', 'כהגדרתה', 'בסעי', 'בה', 'לפקודה', 'העולה', 'על', 'שח', 'מס', 'היה', 'לי', 'לב', 'בת', 'זוגי', 'מחזור', 'מכירות', 'מניירות', 'ער', 'הנסחר', 'בבורסה', 'שאינו', 'פטור', 'ממס', 'העולה', 'על', 'שח', 'תארי', 'הגעה', 'בת', 'הזוג', 'המס', 'היו', 'לי', 'הכנסות', 'מפעילות', 'באינטרנט', 'מסחר', 'שיווק', 'פרסו', 'וכד', 'בבשנת', 'המס', 'היו', 'לי', 'הכנסות', 'בהתא', 'לחוק', 'אנרגיות', 'מתחדשות', 'המס', 'היו', 'לי', 'הכנסות', 'מממוש', 'מטבע', 'ווירטואלי', 'לרבות', 'המרה', 'למטבעות', 'אחרי', 'בני', 'הזוג', 'עיוור', 'או', 'נכה', 'לפי', 'סעי', 'לפקודה', 'בשנת', 'המס', 'העברתי', 'במש', 'שתיי', 'עשרה', 'חודשי', 'כספי', 'אל', 'מחו', 'לישראל', 'בסכו', 'כולל', 'של', 'שח', 'או', 'יותר', 'ספח', 'לחישוב', 'ההכנסה', 'בגי', 'תשלומי', 'עודפי', 'של', 'מעביד', 'לקר', 'השתלמות', 'וקופג', 'טופס', 'שב', 'ישראל', 'מתקימת', 'לגבי', 'חזקת', 'ימי', 'שהייה', 'בישראל', 'הנסתרת', 'על', 'ידי', 'ואני', 'חייב', 'בהגשת', 'דוח', 'לפי', 'סעי', 'לפקודה', 'מצב', 'טופס', 'בנאמנות', 'ברשומה', 'וברחוב', 'דוח', 'זה', 'כולל', 'את', 'הכנסותי', 'ואת', 'הכנסות', 'הנאמנות', 'מצב', 'טופס', 'נה', 'בנאמנות', 'שחלה', 'עליו', 'חובת', 'דיווח', 'לפי', 'סעי', 'אב', 'לפקודה', 'תושב', 'ישראל', 'שמלאו', 'לו', 'עשרי', 'וחמש', 'שנה', 'ושווי', 'נכסי', 'הנאמנות', 'עולה', 'על', 'שח', 'נה', 'בנאמנות', 'שההכנסות', 'שחולקו', 'לי', 'מהנאמנות', 'כלולות', 'בדוח', 'זה', 'מצב', 'העתק', 'טופס', 'נה', 'בנאמנות', 'שממנה', 'היו', 'לי', 'חלוקות', 'פטורותחייבות', 'בשנת', 'המס', 'כהגדרת', 'בסעי', 'הרשומות', 'בדוח', 'זה', 'בשדה', 'מאתיי', 'ושבעי', 'ואחת', 'וזיק', 'זכאי', 'בשותפות', 'נפט', 'ביו', 'ודיווחתי', 'על', 'חלקי', 'בהכנסות', 'השותפות', 'זוגי', 'חבר', 'קיבו', 'מס', 'תיק', 'והכנסה', 'חייבת', 'מועברת', 'לקיבו', 'בהתא', 'לסעי', 'לפקודה', 'הזוג', 'הרשו', 'הזוג', 'גל', 'מניות', 'מהותי', 'בחברת', 'מעטי', 'שחל', 'עליה', 'סעי', 'לפקודה', 'כיר', 'עמי', 'ודיווחתי', 'בהתא', 'לתקנה', 'לפי', 'סעי', 'לפקודה', 'הכנסות', 'מבני', 'לפי', 'סעי', 'אג', 'לפקודה', 'הדות', 'כולל', 'דיווח', 'על', 'סיו', 'בניית', 'פרויקט', 'שתיי', 'מצב', 'טופס', 'שבע', 'מאות', 'ושניי', 'המחזור', 'מכלל', 'שליטה', 'בחבר', 'בני', 'אד', 'תושב', 'חו', 'נסחר', 'בחול', 'מצב', 'טופס', 'מאה', 'וחמישי', 'לא', 'העסקי', 'שלי', 'או', 'של', 'זכויות', 'בחבר', 'בני', 'אד', 'תושב', 'חו', 'שאינו', 'נסחר', 'מצב', 'טופס', 'מאה', 'וחמישי', 'לא', 'בת', 'זוגי', 'הוא', 'מעל', 'אפס', 'היו', 'לי', 'או', 'לב', 'בת', 'זוגי', 'עסקאות', 'צדדי', 'קשורי', 'בחול',
```

'כמשמעות', 'בסעי', 'לפקודה', 'מצב', 'טופס', 'אל', 'שמונה', 'מאות',  
 'ושמונה', 'וחמש', 'לא', 'אפס', 'שיח', 'ולה', 'החייבת', 'בדיווח', 'מכוח',  
 'סעי', 'לפקודה', 'שתי', 'מצב', 'טופס', 'אל', 'מאתי', 'ושלוש', 'עשרה',  
 'לא', 'ללא', 'מעמ', 'ויות', 'דעת', 'חייבת', 'בדיווח', 'המאפשרת', 'יתרו',  
 'מס', 'כאמור', 'בסעי', 'לפקודה', 'שתי', 'מצב', 'טופס', 'אל', 'שלוש',  
 'מאות', 'ארבע', 'וחמש', 'שודר', 'טופס', 'שישה', 'אלפי', 'מאה', 'ואחת',  
 'עשרה', 'מדה', 'חייבת', 'בדיווח', 'הכלולה', 'ברשימה', 'שפרסמה', 'רשות',  
 'המסי', 'כאמור', 'בסעי', 'לפקודה', 'כ', 'מצב', 'טופס', 'אל', 'שמונה',  
 'מאות', 'וארבע', 'ושש', 'לא', 'חייב', 'משק', 'חקלאי', 'לי', 'שטח',  
 'אדמה', 'מעובד', 'מצב', 'טופס', 'באנ', 'או', 'בות', 'זוגי', 'שותפי',  
 'בשותפות', 'מצב', 'טופס', 'נסות', 'מעסקממשת', 'יד', 'עיקרי', 'הדוח',  
 'מבוסס', 'על', 'פנקסי', 'חשבונות', 'שניהלתי', 'עפי', 'תוספת', 'סעי',  
 'להוראות', 'ניהול', 'ספרי', 'עוסק', 'פטור', 'ניהלתי', 'הנהלת', 'חשבונות',  
 'כפולה', 'זחדצידית', 'הפעלתי', 'קופה', 'רושמת', 'לא', 'הפקת', 'תיעוד',  
 'פני', 'זידני', 'ממוחשב', 'ניכוי', 'זיכוי', 'ניכוי', 'מס', 'במקור',  
 'הטבות', 'מס', 'יש', 'לצר', 'מסמכי', 'רלבנטי', 'הזוג', 'הרשו', 'בת',  
 'הזוג', 'ארבעה', 'מספר', 'זהות', 'תשע', 'מאות', 'וארבע', 'מספר', 'זהות',  
 'תשע', 'מאות', 'וארבעעשרה', 'נחתי', 'בשנת', 'המס', 'בשוי', 'מלמד', 'מיכאל',  
 'מלמד', 'ויקטוריה', 'גרוש', 'משפחה', 'פרטי', 'משפחת', 'פרטי', 'איליה',  
 'שלושה', 'מרק', 'בהתא', 'למרש', 'רשות', 'המסי', 'האב', 'תארי', 'לידה',  
 'האב', 'לל', 'מיקוד', 'אחד', 'הע', 'רמת', 'חמישה', 'מיליו', 'מאתי',  
 'ועשרי', 'ואחת', 'אל', 'מאתי', 'ועשרי', 'ושש', 'שבעה', 'ארבעה', 'שבעה',  
 'כתובת', 'למישלות', 'דו', 'שבעה', 'אחד', 'כתובת', 'המגורי', 'וני',  
 'מעוניי', 'לאפשר', 'להשוותהמסי', 'להעביר', 'לי', 'הודעות', 'באמצעות', 'אר',  
 'אלקטרוני', 'רתסוה', 'ימיל', 'לכתובו', 'הדאר', 'האלקטרוני', 'קוו', 'שתי',  
 'עשרה', 'חמישי', 'וארבעה', 'אחד', 'בביתפקס', 'בעבודה', 'הזוג', 'הרשו',  
 'עיקרי', 'פרט', 'אחד', 'סק', 'פר', 'הבית', 'הישוב', 'המיקוד', 'בעסק',  
 'העיקרי', 'ארבעה', 'מספר', 'תיק', 'ניכוי', 'מספר', 'עוסק', 'מעמ',  
 'עבידי', 'מגיע', 'יועבר', 'לחשבונני', 'המתנהלה', 'על', 'שמי', 'בבנק', 'שבעה',  
 'מאתי', 'וחמישי', 'וארבעת', 'אלפי', 'תשע', 'מאות', 'ושלושי', 'ושתי',  
 'שמונה', 'מלמד', 'מיכאל', 'ויקטוריה', 'שוני', 'ואז', 'שינוי', 'פרטי',  
 'חשבו', 'הבנק', 'יש', 'לצר', 'אסמכתא', 'מתאימה', 'מספר', 'חשבו', 'סמל',  
 'סני', 'סמל', 'בנק', 'בעל', 'החשבו', 'כפי', 'שמופיע', 'במרשמי', 'הבנק',  
 'ני', 'מצהיר', 'בזה', 'כי', 'בשנת', 'המס', 'לא', 'היו', 'לי', 'ולב', 'בת',  
 'זוגי', 'הכנסות', 'נוספות', 'על', 'אלו', 'הכלולות', 'בדוח', 'לתשומת',  
 'ליב', 'וכ', 'כי', 'הפרטי', 'שבדי', 'וחשבו', 'זה', 'ונספחיו', 'נכוני',  
 'ומלאי', 'ידוע', 'לי', 'שא', 'המסמכי', 'הוגשו', 'טופס', 'החתו', 'עי',  
 'הזוג', 'הרשו', 'בלבד', 'ללא', 'חתימת', 'הזוג', 'מקוו', 'עלי', 'לשמור',  
 'את', 'המסמכי', 'שצורפו', 'ואת', 'הקבלות', 'המקוריות', 'של', 'התהומות',  
 'השני', 'יראו', 'את', 'החות', 'כמי', 'שהצהיר', 'שבדו', 'ייפוי', 'כוח',  
 'מב', 'זוגי', 'במילוי', 'די', 'וחשבו', 'זה', 'נעזרתי', 'תמורת', 'תשלו',  
 'י', 'על', 'ידי', 'מסייע', 'שפרטיו', 'מצויני', 'להל', 'לחתו', 'בשמו', 'וזאת',  
 'בהתא', 'להוראות', 'סעי', 'מאה', 'וארבע', 'וארבע', 'לפקודת', 'מה',  
 'לידיעת', 'כתובת', 'אתר', 'האינטרנט', 'של', 'רשות', 'המסי', 'בישראל',  
 'אחד', 'אהגז', 'חח', 'תארי', 'חתימת', 'הזוג', 'הרשו', 'חתימת', 'בת',  
 'הזוג', 'המשרד', 'מספר', 'עוסק', 'מורשה', 'מספר', 'טלפו', 'כתובת', 'דואר',  
 'אלקטרוני', 'גיגי', 'ושות', 'רואי', 'חשבו', 'סק', 'חמש', 'שלושה',  
 'החססוחוחהעוז', 'הצ', 'סעי', 'מאה', 'וארבע', 'ושלוש', 'לפקודת', 'מס',  
 'הכנסה', 'אני', 'שפרטי', 'מצויני', 'למעלה', 'מצהיר', 'בזה', 'כי', 'תי',  
 'תמורת', 'תשלו', 'למגיש', 'הדוח', 'בעריכת', 'הדי', 'וחשבו', 'ממ', 'מודע',  
 'לאחריות', 'המוטלת', 'עלי', 'בהקשר', 'זה', 'עפי', 'סעי', 'מאתי', 'ועשרי',  
 'וארבע', 'לפקודת', 'מס', 'הכנסה', 'תארי', 'איש', 'הקשר', 'חתימה', 'ות',  
 'מתחדשות', 'חוק', 'לעידוד', 'השקעה', 'באנרגיות', 'מתחדשות', 'הטבות', 'מס',  
 'בשל', 'הפקת', 'חשמל', 'מאנרגיה', 'מתחדשת', 'התשעז', 'אלפיי', 'ושש',  
 'עשרה', 'כתובת', 'התשעה', 'אלפיי', 'וחמשי', '}]', 'טופס': 16, 'אל': 7, 'שלוש':  
 2, 'מאות': 8, 'ואחד': 1, 'שודר': 2, 'עי': 2, 'מיצג': 1, 'באינטראנט': 1,  
 'אחד': 6, 'מתו': 1, 'ארבעה': 4, 'התשמיס': 1, 'ליחיד': 1, 'ווי': 2,  
 'אסמכתא': 2, 'חותמת': 1, 'המשרד': 2, 'הרשו': 6, 'בלשו': 1, 'זכר': 1,  
 'מתיחס': 1, 'לנקבה': 1, 'ידי': 2, 'וחשבו': 4, 'על': 13, 'הכנסות': 1, 'באר':  
 1, 'ובחול': 1, 'תשע': 5, 'תארי': 6, 'הגשת': 1, 'הדוח': 4, 'ור': 1, 'בשנת':  
 5, 'המס': 9, 'אלפיי': 3, 'ושמונהעשרה': 1, 'שתי': 6, 'השנה': 1, 'המתחילה':

1, 'והמסתיימת': 1, 'השומה': 2, 'תל': 1, 'אביב': 1, 'חמש': 3, 'חוליה': 1, 'שבעה': 5, 'עפי': 3, 'סעי': 14, 'מאה': 6, 'ושלושי': 2, 'ואחת': 4, 'לפקודת': 1, 'מס': 11, 'הכנסה': 6, 'רשאי': 1, 'פקיד': 1, 'לראות': 1, 'מי': 1, 'שהגיש': 1, 'דוח': 6, 'שלא': 3, 'כללי': 1, 'מולא': 1, 'כראוי': 1, 'או': 10, 'יצורפו': 1, 'אליו': 1, 'המסמכי': 3, 'המתאימי': 1, 'כמי': 2, 'הגיש': 1, 'הכנסות': 4, 'והכנסות': 1, 'בת': 13, 'זוגי': 12, 'שלושה': 4, 'בלבד': 2, 'אני': 2, 'מגיש': 2, 'לשנת': 1, 'זו': 1, 'למרות': 1, 'שאיני': 1, 'חייבבקשה': 1, 'להחזיר': 1, 'וגי': 1, 'נפרד': 1, 'מצב': 13, 'הדוחההרה': 1, 'של': 7, 'אי': 1, 'לב': 5, 'עזר': 1, 'לי': 14, 'בהשגת': 1, 'הגעה': 2, 'זוג': 1, 'רשו': 1, 'סה': 1, 'משות': 1, 'לבני': 1, 'הזוג': 12, 'ללא': 3, 'כ': 2, 'עמדת': 2, 'בתנאי': 2, 'לפקודה': 14, 'לא': 8, 'עולה': 2, 'חדשי': 1, 'תושב': 4, 'חוזר': 2, 'ותיק': 1, 'רתושב': 1, 'וחלות': 1, 'מחול': 1, 'הקלות': 1, 'במס': 1, 'ההדהדחק': 1, 'לקדה': 1, 'קמס': 1, 'היו': 7, 'לילדי': 1, 'שטר': 1, 'מלאו': 1, 'לה': 1, 'נכסי': 2, 'בחול': 3, 'בשווי': 1, 'שח': 5, 'יותר': 2, 'היתה': 1, 'חייבת': 4, 'כהגדרתה': 1, 'בסעי': 5, 'בה': 1, 'העולה': 2, 'היה': 1, 'מחזור': 1, 'מכירות': 1, 'מניירות': 1, 'ער': 1, 'הנספחי': 1, 'בבורסה': 1, 'שאינו': 2, 'פטור': 2, 'ממס': 1, 'הכנסות': 6, 'מפעילות': 1, 'באינטרנט': 1, 'מסחר': 1, 'שיווק': 1, 'פרסו': 1, 'וכד': 1, 'בבשנת': 1, 'בהתא': 5, 'לחוק': 1, 'אנרגיות': 1, 'מתחדשות': 3, 'ממימוש': 1, 'מטבע': 1, 'וירטואלי': 1, 'לרבות': 1, 'המרה': 1, 'למטבעות': 1, 'אחרי': 1, 'בני': 3, 'עיוור': 1, 'נכה': 1, 'לפי': 5, 'העברתי': 1, 'במש': 1, 'עשרה': 5, 'חודשי': 1, 'כספי': 1, 'מחו': 1, 'לישראל': 1, 'בסכו': 1, 'כולל': 3, 'ספח': 1, 'לחישוב': 1, 'ההכנסה': 1, 'בגי': 1, 'תשלומי': 1, 'עודפי': 1, 'מעביד': 1, 'לקר': 1, 'השתלמות': 1, 'וקופג': 1, 'שב': 1, 'ישראל': 2, 'מתקימת': 1, 'לגבי': 1, 'חזקת': 1, 'ימי': 1, 'שהייה': 1, 'בישראל': 2, 'הנסתרת': 1, 'ידי': 2, 'ואני': 1, 'חייב': 2, 'בהגשת': 1, 'בנאמנות': 4, 'ברשומה': 1, 'וברחיוב': 1, 'זה': 6, 'את': 3, 'ואת': 2, 'הנאמנות': 2, 'נה': 3, 'שחלה': 1, 'עליו': 1, 'חובת': 1, 'דיווח': 2, 'אב': 1, 'שמלאו': 1, 'לו': 1, 'עשרי': 1, 'וחמש': 4, 'שנה': 1, 'ושווי': 1, 'שההכנסות': 1, 'שחולקו': 1, 'מהנאמנות': 1, 'כלולות': 1, 'בדוח': 3, 'העתק': 1, 'שממנה': 1, 'חלוקות': 1, 'פטורותחייבות': 1, 'כהגדרת': 1, 'הרשומות': 1, 'בשדה': 1, 'מאתי': 6, 'ושבעי': 1, 'וזיק': 1, 'זכאי': 1, 'בשותפות': 2, 'נפט': 1, 'ביו': 1, 'ודיווחתי': 2, 'חלקי': 1, 'בהכנסות': 1, 'השותפות': 1, 'חבר': 1, 'קייבו': 1, 'תיק': 2, 'והכנסה': 1, 'מועברת': 1, 'לקייבו': 1, 'לסעי': 1, 'גל': 1, 'מניות': 1, 'מהותי': 1, 'בחברת': 1, 'מעטי': 1, 'שחל': 1, 'עליה': 1, 'כיר': 1, 'עמי': 1, 'לתקנה': 1, 'מבני': 1, 'אג': 1, 'הדות': 1, 'סיו': 1, 'בניית': 1, 'פרויקט': 1, 'שבע': 1, 'ושניי': 1, 'המחזור': 1, 'מכלל': 1, 'שליטה': 1, 'בחבר': 2, 'אד': 2, 'חו': 2, 'נסחר': 2, 'וחמישי': 3, 'העסקי': 1, 'שלי': 1, 'זכויות': 1, 'הוא': 1, 'מעל': 1, 'אפס': 2, 'עסקאות': 1, 'צדדי': 1, 'קשורי': 1, 'כמשמעות': 1, 'שמונה': 3, 'ושמוני': 1, 'שיח': 1, 'ולה': 1, 'החייבת': 1, 'בדיווח': 3, 'מכוח': 1, 'ושלוש': 2, 'מעמ': 1, 'ווות': 1, 'דעת': 1, 'המאפשרת': 1, 'יתרו': 1, 'כאמור': 2, 'ארבעי': 1, 'שישה': 1, 'אלפי': 2, 'מדה': 1, 'הכלולה': 1, 'ברשימה': 1, 'שפרסמה': 1, 'רשות': 3, 'המסי': 2, 'וארבעי': 3, 'ושש': 3, 'משק': 1, 'חקלאישי': 1, 'שטח': 1, 'אדמה': 1, 'מעובד': 1, 'באנ': 1, 'בות': 1, 'שותפי': 1, 'נסות': 1, 'מעסקממשלת': 1, 'ידי': 1, 'עיקרי': 2, 'מבוסס': 1, 'פנקסי': 1, 'חשבונות': 2, 'שניהלתי': 1, 'תוספת': 1, 'להוראות': 2, 'ניהול': 1, 'ספרי': 1, 'עוסק': 3, 'ניהלתי': 1, 'הנהלת': 1, 'כפולה': 1, 'זחזחידית': 1, 'הפעלתי': 1, 'קופה': 1, 'רושמת': 1, 'הפקת': 2, 'תיעוד': 1, 'פני': 1, 'זידיני': 1, 'ממוחשב': 1, 'ניכויי': 2, 'זיכויי': 1, 'ניכוי': 1, 'במקור': 1, 'הטבות': 2, 'יש': 2, 'לצר': 2, 'מסמכי': 1, 'רלבנטיי': 1, 'מספר': 7, 'זהות': 2, 'וארבע': 3, 'וארבעעשרה': 1, 'נחתי': 1, 'בשווי': 1, 'מלמד': 3, 'מיכאל': 2, 'ויקטוריה': 1, 'גרוש': 1, 'משפחה': 1, 'פרטי': 3, 'משפחת': 1, 'איליה': 1, 'מרק': 1, 'למרש': 1, 'האב': 2, 'לידה': 1, 'לל': 1, 'מיקוד': 1, 'הע': 1, 'רמת': 1, 'חמישה': 1, 'מיליו': 1, 'ועשרי': 3, 'כתובת': 5, 'למישלות': 1, 'דו': 1, 'המגורי': 1, 'וני': 1, 'מעוניי': 1, 'לאפשר': 1, 'להשוותהמסי': 1, 'להעביר': 1, 'הודעות': 1, 'באמצעות': 1, 'אר': 1, 'אלקטרוני': 1, 'רתסוה': 1, 'ימייל': 1, 'לכתובו': 1, 'הדאר': 1, 'האלקטרוני': 1, 'קוו': 1, 'חמישי': 1, 'וארבעה': 1, 'בביתפקס': 1, 'בעבודה': 1, 'פרט': 1, 'סק': 2, 'פר': 1, 'הבית': 1, 'הישוב': 1, 'המיקוד': 1, 'בעסק': 1, 'העיקרי': 1, 'במעמ': 1, 'עבידי': 1, 'מגיע': 1, 'יועבר': 1, 'לחשבונ': 1, 'המתנהל': 1, 'שמי': 1, 'בבנק': 1, 'וארבעת': 1, 'ושתיי': 1, 'וויקטוריה': 1, 'שוני': 1, 'ואז': 1, 'שינוי': 1,



'חשבו': 3, 'הבנק': 2, 'מתאימה': 1, 'סמל': 2, 'סני': 1, 'בנק': 1, 'בעל': 1,  
'החשבו': 1, 'כפי': 1, 'שמופיע': 1, 'במרשמי': 1, 'ני': 1, 'מצהיר': 2, 'בזה':  
2, 'כי': 3, 'ולב': 1, 'נוספות': 1, 'אלו': 1, 'הכלולות': 1, 'לתשומת': 1,  
'ליב': 1, 'וכ': 1, 'הפרטי': 1, 'שבדי': 1, 'יונספחיו': 1, 'נכוני': 1,  
'ומלאי': 1, 'ידוע': 1, 'שא': 1, 'הוגשו': 1, 'החתו': 1, 'חתימת': 3, 'מקוו':  
1, 'עלי': 2, 'לשמור': 1, 'שצורפו': 1, 'הקבלות': 1, 'המקוריות': 1,  
'התהומות': 1, 'השני': 1, 'יראו': 1, 'החות': 1, 'שהצהיר': 1, 'שבידו': 1,  
'ייפוי': 1, 'כוח': 1, 'מב': 1, 'זוגו': 1, 'במילוי': 1, 'נעזרתי': 1,  
'תמורת': 2, 'תשלו': 2, 'מסייע': 1, 'שפרטיו': 1, 'מצויני': 1, 'להל': 1,  
'לחתו': 1, 'בשמו': 1, 'וזאת': 1, 'מה': 1, 'לידיעת': 1, 'אתר': 1,  
'האינטרנט': 1, 'המיסי': 1, 'אהגז': 1, 'חח': 1, 'מורשה': 1, 'טלפו': 1,  
'דואר': 1, 'אלקטרוני': 1, 'גיגי': 1, 'יושות': 1, 'רואי': 1, 'החססווחהעוז':  
1, 'הצ': 1, 'שפרטי': 1, 'מצויני': 1, 'למעלה': 1, 'תי': 1, 'למגיש': 1,  
'בעריכת': 1, 'הדי': 1, 'ממ': 1, 'מודע': 1, 'לאחריות': 1, 'המוטלת': 1,  
'בהקשר': 1, 'איש': 1, 'הקשר': 1, 'חתימה': 1, 'ות': 1, 'חוק': 1, 'לעידוד': 1,  
'השקעה': 1, 'באנרגיות': 1, 'בשל': 1, 'חשמל': 1, 'מאנרגיה': 1, 'מתחדשת': 1,  
1, 'התשעז': 1, 'התשסה': 1, '}]

## ספר 2: ספר על קוד ריצה במקביל

ספר זה אנו נסביר את האלגוריתמיקה של התיכנות במקביל שלנו ומה כל פונקציה עושה. ניתן להריץ את הפונקציות אחד אחד בשביל לראות את התוצאות לשהם וזאת ספר זה אנו נסביר את האלגוריתמיקה של התיכנות במקביל שלנו ומה כל פונקציה עושה. ניתן להריץ את הפונקציות אחד אחד בשביל לראות את התוצאות לשהם וזאת

### Import:

בחלק זה אנחנו טוענים את כל הסיפריים והמשתנים הגלובים השתמשנו כדי ליצור את המערכת ריצת קוד במקביל

In [2]:

```
import os #סיפריה אשר נותנת לנו פעולות של מערכת הפעלה
import multiprocessing #סיפריה שנותמת לנו ליצור פרוסס חדש במערכת הפעלה
import time #סיפריה שנותנת לנו אפשרות לבצע פעולות על זמן
import numpy as np #סיפריה הרחבה לפעולות מתמטיקאיות על מערכים וכ'ו
import threading #סיפריה אשר נותנת לנו לעשות פעולות אם תהליכונים
import pickle #סיפריה עזר ליצירת קבצים
import Pdf2Text #פעולות של ספר מספר 1
import heapq #סיפריה של פעולות על תור עדיפויות
import config #דף הגדרות של הפרויקט
theadcount = 0 #משתנה גלובלי אשר סופר כמה תהליכונים פעילים
processBagofwords = {} #משתנה גלובלי אשר מחזיק את רשימת המילים הכי שכיחות
#בתהליך מסויים
prossesstokenarry = [] #משתנה גלובלי אשר מחזיק את המידע התקבל אחרי הפעולות
#יצירת ווקטור מילים שקורא בתהליך מסויים
```

### תהליך: יצירת תהליכונים קטנים

הפעולה מקבלת מערך של שלמות קבצים, כתובת התקיה שהקבצים נמצאים בא, מליון תהליכים, ומפתח תהליכים

procnum, return\_dict, root, files

הפעולה מחקלת את המערך הקבצים לעשרה מערכים שונים. ויוצרת תהליכונים

הפעולה מחזירה לתוך מילון התהליך את רשימה שהתקבלה אחרי תהליך המתבצ בספר 1. ווקטור מילים המסודרים לפי המילה הכי שכיחה להכי נדירה

In [3]:

```
def process(files, root, return_dict, procnum):
    if len(files) > 0:
        Bagofwords = {}
        theadsarry = []
        tokenarry = []
        global processBagofwords
        global prossesstokenarry
        processlock = threading.Lock()
        global theadcount
        splittedfiles = np.array_split(files, 10)
        for newfiles in splittedfiles: #לולאה היוצרת תהליכון לכל מערך
            t1 = threading.Thread(target=Theard,
            args=(root, newfiles, processlock,))
            theadsarry.append(t1)
            theadcount+=1
            time.sleep(0.5)
        for t in theadsarry: #לולאה אשר מפעילה את המערך
            t.start()
        while theadcount > 0: #לולאה אשר בודקת אם כל התהליכונים עדיין בחיים
            if len(processBagofwords) > 0: #בדיקה עם יש מילים בתוך ווקטור
                processlock.acquire()
                Bagofwords = Pdf2Text.margedictionaries(Bagofwords, processBagofwords)
```

```

        processBagofwords= {}
        processlock.release()
        if (len(prosesselokenarry)>0): # בד"קה אם יש את הרישמה הנוצרת
בספר 1
            processlock.acquire()
            tokenarry += prosesselokenarry
            prosesselokenarry=[]
            processlock.release()
        return_dict[procnum] = [tokenarry,Bagofwords]

```

## ויצרת ווקטור מילים pdf תהליכון: המרת/נקוי המילים מהקבוצ

הפעולה מקבלת מערך של שלמות קבצים, כתובת התקליה שהקבצים נמצאים בא, מפתח לסמפור

root,files,processlock

פעולה מפעילה את פעולת המרת המילים מספר 1 תוך כדי סיכרון עם כל התהליכונים האחרים

הפעולה מחזירה את הרשימה שהתקבלה בספר 1 ואת הרשימת המילים השכיחה ביותר בתהליכון המסויים זה.

In [4]:

```

def Theard(root,files,processlock,):
    global processBagofwords
    global prosesselokenarry
    global theadcount
    thearbagofword = {}
    tokenarry = []
    try:
        for name in files: # לולאה שרצה על כל המסמכים ויוצרת האת המודל של
ספר 1
            bagofword =
Pdf2Text.BuildModelToTraing(os.path.join(os.path.join(root, name)), name,
processlock, ) #2 פעולה מספר
            thearbagofword = Pdf2Text.margedictionaries(thearbagofword,
bagofword[1])
            tokenarry.append(bagofword[0])
            processlock.acquire() #סמפור
            processBagofwords = Pdf2Text.margedictionaries(processBagofwords,
thearbagofword)
            prosesselokenarry += tokenarry
            theadcount -= 1
            processlock.release()
    except Exception as err:
        processlock.acquire()
        theadcount -= 1
        processlock.release()
        raise err

```

## יצירת תהליכים

הפעולה מקבלת את מספר הליבות במעמד, מיקום התקיות, רפרנס למספר הפרוסס, מילון החזרה של תהליך, ומערך של תהליכים פעילים

פעולה הזאת יוצרת את התהליכים החדשים לפי מספר הליבות במערכת בשביל לא ליצור עומס

In [5]:

```
def Startingproses(cpu_count, dircount, prossesid, return_dict,
theardonjobnow):
    for baseroot, directories, nonfiles in
os.walk(os.path.join(os.path.join(config.WORKSPACE,config.TRANING)),
topdown=False):
        #לולא שבודקת את התקיות שהוגדר באזור העבודה
        allfiles = []
        for root, dirs, files in os.walk(baseroot):# קובץ על כל
            #קובץ לתוך מערך קבצים של כל
            for f in files:
                allfiles.append(f)
            splitefile = np.array_split(files, cpu_count / dircount)# פעולה
            #מפצלת את המערך למערכים קטנים לפי כמות הליבות
            for filestosend in splitefile: #הולאה יוצרת תהליך חדש
                if (baseroot ==
(os.path.join(config.WORKSPACE,config.TRANING))):
                    p1 = multiprocessing.Process(target=process,
args=(filestosend, root, return_dict, prossesid))
                    theardonjobnow.append(p1)
                    prossesid = prossesid + 1
            for p in theardonjobnow:#הפעולה מפעילה את כל התהליכים שנוצרו
                p.start()
```

## ניהול התהליכים

פעולה זאת מקבלת מערך ריק של המילים הכי שכיחות, רשימה ריקה של התהליך שקורה בספר I, מספר הליבות במעמד, מיקום התקיות, רפרנס למספר הפרוסס, מילון החזרה של תהליך, ומערך של תהליכים פעילים

הפעולה יוצרת ונהלת את על התהליכים עד שהם מסתיימים. ושומרת בקובץ את המערך המילים השייכות ואת מערך הרשימות של ספר I

In [10]:

```
def BuildBagOfWord(Bagofwords, cpu_count, dircount, i, return_dict,
theardonjobnow, tokenarry):
    for baseroot, directories, nonfiles in
os.walk(os.path.join(os.path.join(config.WORKSPACE, config.TRANING)),
topdown=False):
        #לולאה בודק איזה סוג מסמך זה על פי שם תקייה
        if len(directories) > dircount:
            dircount = len(directories)
        Startingproses(cpu_count, dircount, i, return_dict, theardonjobnow) #
הפעולה יוצרת תהליכים
        while (len(theardonjobnow) > 0): # כל התהליכים עד
שהם מסתיימים
            for poress in theardonjobnow: # כל התהליכים בחיים
אחרת מוציאה אותו מהמערך
                if poress.is_alive() == False:
                    theardonjobnow.remove(poress)
            time.sleep(0.2)
            for x in return_dict: # לולאה לוחקת את מה שהוחזר המפרוסס ומאחדת אותם
למערכים שלמים אחד של במילים ואחד של הרשימות
                Bagofwords = Pdf2Text.margedictionaries(return_dict[x][1],
Bagofwords)
                tokenarry += return_dict[x][0]
            most_freq = heapq.nlargest(900, Bagofwords, key=Bagofwords.get) # יציאת
ערמה של המלים רק ל 900 המלילים הכי שכיחות מהמליון שכיכיות
            file = open(os.path.join(config.WORKSPACE, config.BAGOFWORDSFILE), 'wb')
            pickle.dump(most_freq, file)
            file = open(os.path.join(config.WORKSPACE, config.TOKENTOTRAINFILE),
'wb')
            pickle.dump(tokenarry, file)
            #יצירת template יצירת שני קבצים של ערמת שכיכיות שאחר כך תהפוך
            #יצירת הרשימה של ספר 1 בשביל להמשיך התהליך של לימוד מוכנה
```

## למכונה input הוספת

הפעולה הזאת מקבלת את הרשימה מספר 1 ומוסיפה קלט למכונה

In [7]:

```
def GetDataFromFile(tokenototrain):
    try:
        traningset = []
        file = open(os.path.join(config.WORKSPACE, config.BAGOFWORDSFILE),
            'rb')
        Bagofwords = pickle.load(file)
        file.close()
        if (tokenototrain == None):
            file = open(os.path.join(config.WORKSPACE,
config.TOKENTOTRAINFILE), 'rb')
            tokenototrain = pickle.load(file)
            file.close()
            #פתחת את הקבצים שיצרנו בפעולה הקודמת
        for file in tokenototrain: #הולאה רוצה על כל ה
            data = file
            sent_vec = []
            for word in Bagofwords:
                if word in file[2]: # התנאי בודק את המילים שבטקסט ובודק עם
                    #הנמצאים מסנן אותם באחד במקום של המילה אחרת נסמן ב0
                    sent_vec.append(1)
                else:
                    sent_vec.append(0)
            sent_vec = np.asarray(sent_vec)
            data.append(sent_vec)
            traningset.append(data)
        return (Bagofwords, traningset)
    except:
        return config.ERRORLABEL
```

## פונקציה לחישוב מקבלי

הפונקציה הזאת מחברת את כל הפונקציות שיצרנו ויצירת מהם מאגר של מילים שחיכות וגם יוצרת רשימה של ספר 1 עם אינפוט למכונה.

שני הפונקציות עושות אותו הדבר אבל אחת יש פרמטר שאומר את המיקום ועושה את הפעולה הזאת אך ללא שמומש בתהליכים אלא רק בתהליכים

In [8]:

```
def begofword():
    cpu_count = multiprocessing.cpu_count()
    Bagofwords = {}
    tokenarry = []
    theardonjobnow = []
    manager = multiprocessing.Manager()
    return_dict = manager.dict()
    dircount = 0
    i = 0
    if(not os.path.exists(os.path.join(config.WORKSPACE,
config.BAGOFWORDSFILE))):
        BuildBagOfWord(Bagofwords, cpu_count, dircount, i, return_dict,
theardonjobnow, tokenarry)
        return GetDataFromFile(None)

def bagofwordonlytheard(when):
    Bagofwords = {}
    theadsarry = []
    tokenarry = []
    global processBagofwords
    global prossesstokenarry
    global theadcount
    processlock = threading.Lock()
    allfiles = []
    for root, directories, files in os.walk(when, topdown=False):
        allfiles += (files)
    if len(allfiles) > 2:
        splitedfilsses = np.array_split(allfiles, len(allfiles) / 2)
        for newfiles in splitedfilsses:
            t1 = threading.Thread(target=Theard, args=(root, newfiles,
processlock,))
            theadsarry.append(t1)
            theadcount += 1
    else:
        splitedfilsses =allfiles
        t1 = threading.Thread(target=Theard, args=(root, splitedfilsses,
processlock,))
        theadsarry.append(t1)
        theadcount += 1
    time.sleep(0.5)
    for t in theadsarry:
        t.start()
    while (theadcount > 0):
        if (len(processBagofwords) > 0):
            processlock.acquire()
            Bagofwords = Pdf2Text.margedictionaries(Bagofwords,
processBagofwords)
            processBagofwords = {}
            processlock.release()
        if (len(prossesstokenarry) > 0):
            processlock.acquire()
            tokenarry += prossesstokenarry
            prossesstokenarry = []
```

processlock.release()

**return** GetDataFromFile(tokenarry)[1]

אם נילחץ על הקוד כאן למטה יהיה אפשר לראות את מה מחזירה הפונקציה

לפעולה הזאת לוקחת החן 1-5 דקות

In [9]:

```
import warnings
warnings.filterwarnings("ignore")
print(bagofwordonlyheard(os.path.join("example")))
טופס, אל, שלוש, מאות, ואחד, [ 'example\\2015.pdf', [ 'לבדיקה', 'התשמיס', 'שודר', 'עי', 'מייצג', 'באינטראנט', 'אחד', 'מתו', 'ארבעה', 'התשמיס', 'ליחיד', 'וו', 'וו', 'אסמכתא', 'חותמת', 'המשרד', 'הרשו', 'בלשו', 'זכר', 'מתיחס', 'לנקבה', 'די', 'וחשבו', 'על', 'ההכנסות', 'באר', 'ובחול', 'תשע', 'תשע', 'תארי', 'הגשת', 'הדוח', 'ור', 'הדוח', 'בשנת', 'המס', 'אלפיי', 'ושמונהעשרה', 'שתיי', 'השנה', 'המתחילה', 'והמסתיימת', 'השומה', 'תל', 'אביב', 'חמש', 'חוליה', 'שבעה', 'עפי', 'סעי', 'מאה', 'ושלושי', 'ואחת', 'לפקודת', 'מס', 'הכנסה', 'רשאי', 'פקיד', 'השומה', 'לראות', 'מי', 'שהגיש', 'דוח', 'שלא', 'כללי', 'מולא', 'כראוי', 'או', 'שלא', 'צורפו', 'אליו', 'המסמכי', 'המתאימי', 'כמי', 'שלא', 'הגיש', 'דוח', 'על', 'הכנסות', 'והכנסות', 'בת', 'זוגי', 'שלושה', 'הכנסות', 'בלבד', 'חמש', 'אני', 'מגיש', 'דוח', 'לשנת', 'מס', 'זו', 'למרות', 'שאיני', 'חייבבקשה', 'להחזיר', 'מס', 'וגי', 'מגיש', 'דוח', 'נפרד', 'מצב', 'הדוחהצהרה', 'של', 'בת', 'זוגי', 'אי', 'הכנסה', 'לב', 'בת', 'זוגי', 'בת', 'זוגי', 'עזר', 'לי', 'בהשגת', 'הכנסה', 'תארי', 'הגעה', 'זוג', 'רשו', 'סה', 'משות', 'לבני', 'הזוג', 'ללא', 'כי', 'עמדת', 'בתנאי', 'סעי', 'לפקודה', 'שלושה', 'לא', 'עמדת', 'בתנאי', 'סעי', 'לפקודה', 'עולה', 'חדש', 'תושב', 'חוזר', 'ותיק', 'רתושב', 'חוזר', 'וחלות', 'על', 'הכנסות', 'מחול', 'הקלות', 'במס', 'ההדחקה', 'לקדה', 'קמס', 'היו', 'לי', 'או', 'לב', 'בת', 'זוגי', 'או', 'לילדי', 'שטר', 'מלאו', 'לה', 'נכסי', 'בחול', 'בשווי', 'של', 'שח', 'או', 'יותר', 'המס', 'היתה', 'לי', 'או', 'לב', 'בת', 'זוגי', 'הכנסה', 'חייבת', 'כהגדרתה', 'בסעי', 'בה', 'לפקודה', 'העולה', 'על', 'שח', 'מס', 'היה', 'לי', 'לב', 'בת', 'זוגי', 'מחזור', 'מכירות', 'מניירות', 'ער', 'הנסחר', 'בבורסה', 'שאינו', 'פטור', 'ממס', 'העולה', 'על', 'שח', 'תארי', 'הגעה', 'בת', 'הזוג', 'המס', 'היו', 'לי', 'הכנסות', 'מפעילות', 'באינטרנט', 'מסחר', 'שיווק', 'פרסו', 'וכד', 'בבשנת', 'המס', 'היו', 'לי', 'הכנסות', 'בהתא', 'לחוק', 'אנרגיות', 'מתחדשות', 'המס', 'היו', 'לי', 'הכנסות', 'מממוש', 'מטבע', 'ווירטואלי', 'לרבות', 'המרה', 'למטבעות', 'לפקודה', 'אחרי', 'בני', 'הזוג', 'עיוור', 'או', 'נכה', 'לפי', 'סעי', 'בשנת', 'המס', 'העברתי', 'במש', 'שתיי', 'עשרה', 'חודשי', 'כספי', 'אל', 'מחו', 'לישראל', 'בסכו', 'כולל', 'של', 'שח', 'או', 'יותר', 'ספח', 'לחישוב', 'ההכנסה', 'בגי', 'תשלומי', 'עודפי', 'של', 'מעביד', 'לקר', 'השתלמות', 'וקופג', 'טופס', 'שב', 'ישראל', 'מתקיימת', 'לגבי', 'חזקת', 'ימי', 'שהייה', 'בישראל', 'הנסתרת', 'על', 'ידי', 'ואני', 'חייב', 'בהגשת', 'דוח', 'לפי', 'סעי', 'לפקודה', 'מצב', 'טופס', 'בנאמנות', 'ברשומה', 'וברחיוב', 'דוח', 'זה', 'כולל', 'את', 'הכנסות', 'ואת', 'הכנסות', 'הנאמנות', 'מצב', 'טופס', 'נה', 'בנאמנות', 'שחלה', 'עליו', 'חובת', 'דיווח', 'לפי', 'סעי', 'אב', 'לפקודה', 'תושב', 'ישראל', 'שמלאו', 'לו', 'עשרי', 'וחמש', 'שנה', 'ושווי', 'נכסי', 'הנאמנות', 'עולה', 'על', 'שח', 'נה', 'בנאמנות', 'שהכנסות', 'שחולקו', 'לי', 'מהנאמנות', 'כלולות', 'בדוח', 'זה', 'מצב', 'העתק', 'טופס', 'נה', 'בנאמנות', 'שממנה', 'היו', 'לי', 'חלוקות', 'פטורותחייבות', 'בשנת', 'המס', 'כהגדרת', 'בסעי', 'הרשומות', 'בדוח', 'זה', 'בשדה', 'מאתי', 'ושבעי', 'ואחת', 'וזיק', 'זכאי', 'בשותפות', 'נפט', 'ביו', 'ודיווחתי', 'על', 'חלקי', 'בהכנסות', 'השותפות', 'זוגי', 'חבר', 'קיבו', 'מס', 'תיק', 'והכנסה', 'חייבת', 'מועברת', 'לקיבו', 'בהתא', 'לסעי', 'לפקודה', 'הזוג', 'הרשו', 'הזוג', 'גל', 'מניות', 'מהותי', 'בחברת', 'מעטי', 'שחל', 'עליה', 'סעי', 'לפקודה', 'כיר', 'עמי', 'ודיווחתי', 'בהתא', 'לתקנה', 'לפי', 'סעי', 'לפקודה', 'הכנסות', 'מבני', 'לפי', 'סעי', 'אג', 'לפקודה', 'הדות', 'כולל', 'דיווח', 'על', 'סיו', 'בניית', 'פרויקט', 'שתיי', 'מצב', 'טופס', 'שבע', 'מאות', 'ושניי', 'המחזור', 'מכלל', 'שליטה', 'בחבר', 'בני', 'אד', 'תושב', 'חו', 'נסחר', 'בחול', 'מצב', 'טופס', 'מאה', 'וחמישי', 'לא', 'העסקי', 'שלי', 'או', 'של', 'זכויות', 'בחבר', 'בני', 'אד', 'תושב', 'חו', 'שאינו', 'נסחר',
```



65 סיווג מסמכים אוטומטי | לב יודצ'ב \ עמרי גיגי

```
1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0,
0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0,
1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 0,
1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1,
1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1,
1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1,
1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1,
1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 1,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 0,
1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0,
1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]]
```

In [8]:

```
import os
import config
import pickle
file = open(os.path.join(config.WORKSPACE, config.BAGOFWORDSFILE), 'rb')
Bagofwords = pickle.load(file)
file.close()
print(Bagofwords)
['על', 'טופס', 'בת', 'מספר', 'סעי', 'מצב', 'אל', 'מאות', 'לי', 'לפקודה', 'המס', 'זוגי', 'אחד', 'של', 'או', 'עשרה', 'הזוג', 'בשנת', 'מס', 'תארי', 'רכוש', 'זה', 'מאה', 'הנני', 'חמש', 'הכנסה', 'לפי', 'מבוסס', 'לא', 'דוח', 'שלושה', 'כתובת', 'מאתי', 'עשרי', 'את', 'שמונה', 'עלי', 'היו', 'שתי', 'שלוש', 'וחמש', 'שבע', 'הדוח', 'הכנסות', 'הרשו', 'ארבע', 'תושב', 'תיק', 'סהיכ', 'ארבעי', 'לפקודת', 'כולל', 'לב', 'בסעי', 'וחשבו', 'אפס', 'שש', 'שח', 'עוסק', 'מאז', 'חשבו', 'קבוע', 'פרטי', 'וארבע', 'בהתא', 'ואחת', 'מיליו', 'בנאמנות', 'ושש', 'הכנסות', 'עפי', 'חייבת', 'תשע', 'לשותפות', 'נער', 'עד', 'ספירה', 'הערכה', 'שוט', 'הפרטי', 'בעל', 'בעסק', 'קופה', 'ושמונה', 'יש', 'די', 'בשותפות', 'אני', 'שלא', 'בלבד', 'חשבונות', 'במעמ', 'יותר', 'אלפיי', 'לצר', 'ושלושי', 'נהנה', 'כי', 'המסמכי', 'המסי', 'בדיווח', 'מי', 'שאינו', 'חתימת', 'ושלוש', 'רשות', 'בדוח', 'בחול', 'מתחדשות', 'החשבו', 'המשרד', 'דואר', 'חוק', 'הו', 'ושני', 'ניכוי', 'ותשע', 'חמישי', 'אחר', 'מלאי', 'לכל', 'חלק', 'זוג', 'הטופס', 'שישי', 'להמציא', 'למשרדי', 'והמחאות', 'כה', 'ובנקיס', 'שבעי', 'הבנק', 'גיגי', 'לה', 'ושבע', 'תשלומי', 'באר', 'שמוני', 'יד', 'מלאו', 'שנה', 'חוליה', 'נספח', 'וחמישי', 'ושות']
```

'רואי', 'אד', 'מייצג', 'חו', 'מתו', 'ובחול', 'בני', 'ואת', 'השומה', 'כמי',  
'בזבר', 'להוראות', 'מבני', 'כאמור', 'ישראל', 'חותמת', 'עולה', 'העולה',  
'טלפו', 'מסמכי', 'מגיש', 'עמדת', 'העיקרי', 'העסק', 'בזה', 'בתנאי', 'לגבי',  
'תשלו', 'ההכנסות', 'תו', 'לו', 'הפקת', 'סמל', 'ועשרי', 'עי', 'נסחר',  
'ניהלתי', 'לרבות', 'בעבודה', 'תמורת', 'חוזר', 'לידה', 'ידי', 'נכסי',  
'דיווח', 'מצהיר', 'הנאמנות', 'משפחה', 'אנרגיות', 'ללא', 'תל', 'פטור',  
'האב', 'הטבות', 'אלקטרוני', 'זהות', 'בישראל', 'אביב', 'ודיווחתי', 'פרוד',  
'הגעה', 'שודר', 'הוא', 'אסמכתא', 'הגשת', 'ואו', 'ואחד', 'אחת', 'וו', 'בו',  
'חייב', 'גוש', 'שנת', 'חיי', 'יו', 'משרד', 'לרשו', 'רכב', 'שלושי', 'בטופס',  
'חתו', 'זו', 'סוג', 'גל', 'דמי', 'בשורה', 'וציוד', 'שות', 'הצהרת', 'שומה',  
'טמל', 'למלא', 'מוחזקת', 'לציי', 'זכות', 'בצירו', 'אישור', 'טר', 'אלו',  
'צר', 'הער', 'הנ', 'נכס', 'פיישי', 'מעסק', 'רב', 'וריהוט', 'אלפי', 'לכבוד',  
'דג', 'הדפסה', 'דרישה', 'בתוק', 'סמכות', 'אבקש', 'המצב', 'הרכוש',  
'וההתחייבות', 'וילדי', 'שבשנת', 'לאימות', 'המוצהרי', 'מיו', 'קבלתו',  
'כשהוא', 'ממלא', 'פרטיו', 'וחתו', 'הסבר', 'למילוי', 'בנספח', 'המצור',  
'בכבוד', 'וההתחייבות', 'ליוס', 'מושקע', 'בעסקים', 'בזלק', 'המתייחס',  
'בכללה', 'כאילו', 'באחוזי', 'ובסכו', 'ולצר', 'וההתחייבות', 'ברשות',  
'מהעסקי', 'חלקה', 'רכישה', 'מגרשי', 'בניינים', 'מטעי', 'שטחי', 'מורע',  
'חקלאיסחממותבעלי', 'ראויה', 'מוניטי', 'רכישת', 'כבד', 'יצורמספר', 'מוצרי',  
'חומרי', 'עבודות', 'בביצוע', 'שטרות', 'מעטדות', 'ולקוחות', 'רשימה',  
'וכתובת', 'מטייח', 'עסקי', 'הסוג', 'השער', 'וסהייכ', 'בשיח', 'ובמטי',  
'בקופה', 'לזיהוי', 'הופק', 'עיי', 'יהצהרונית', 'בתארי', 'בשקליס', 'חדשי',  
'מסהכנסה', 'נשוי', 'במבנים', 'מפתכתובת', 'פא', 'שיפורי',  
'מכונותצידמכשירים', 'רשול', 'שא', 'חייבים', 'מזומנים', 'בה', 'שליטה', 'כ',  
'בגי', 'פעולה', 'וכ', 'נפרד', 'ההכנסה', 'כלולות', 'אג', 'מכוח', 'שבדי',  
'שטר', 'לשנת', 'אי', 'משות', 'לחישוב', 'שממנה', 'מכלל', 'צדדי', 'קשורי',  
'מעובד', 'הריני', 'ולב', 'נוספות', 'הכלולות', 'נכוני', 'מצוייני', 'פקיד',  
'צורפו', 'היה', 'עיוור', 'נכה', 'לחוק', 'עסקאות', 'קיימת', 'דעת', 'בעמדה',  
'ברשימה', 'ניהול', 'ספרי', 'משק', 'שטח', 'אדמה', 'השני', 'החות', 'מב',  
'לאחריות', 'באינטרנט', 'השנה', 'רשאי', 'מולא', 'למרות', 'ותיק', 'מעביד',  
'עליו', 'חלוקות', 'כהגדרת', 'ביו', 'המחזור', 'זכויות', 'יתרו', 'הכלולה',  
'שפרסמה', 'הנהלת', 'פני', 'שותפי', 'הישוב', 'המעבידי', 'לשמור', 'שצורפו',  
'הקבלות', 'התרומות', 'שבדיו', 'במילוי', 'למעלה', 'בעריכת', 'המוטלת',  
'בהקשר', 'אב', 'לראות', 'כללי', 'והכנסות', 'שאיני', 'רשו', 'לבני', 'ממס',  
'מחזור', 'במש', 'חודשי', 'מפעילות', 'העתק', 'שיש', 'כמשמעות', 'החייבת',  
'המאפשרת', 'תוספת', 'הפעלתי', 'המגור', 'העיסוק', 'פרט', 'ונספחיו',  
'הוגשו', 'שהצהיר', 'שפרטי', 'הקלות', 'מכירות', 'ער', 'בלשו', 'שהגיש',  
'הגיש', 'בהגת', 'מחו', 'לישראל', 'השתלמות', 'יוצר', 'ברשומה', 'שההכנסות',  
'הרשומות', 'קייבלתי', 'חקלאישי', 'משפחתי', 'הרשוב', 'שמות', 'ידוע', 'החתו',  
'מקוו', 'יראו', 'נעזרתי', 'בשמו', 'בשל', 'וקופג', 'כהגדרתה', 'עודפי',  
'לקר', 'תשעי', 'זכר', 'מתיחס', 'המתחילה', 'והמסתיימת', 'המתאימי',  
'הדוחהצהרה', 'מקור', 'אניב', 'חדש', 'בבורסה', 'כספי', 'מסחר', 'וכד',  
'חזקת', 'שמלאו', 'בשדה', 'מחזיק', 'זכאי', 'נפט', 'סיו', 'פנקסי', 'תיעוד',  
'שהוא', 'טלפוני', 'באופ', 'המקוריות', 'להל', 'מה', 'מורשה', 'סייעתי',  
'למגיש', 'מודע', 'שיווק', 'בהגשת', 'וחלות', 'שהייה', 'הנסתרת', 'שכיר',  
'כראוי', 'אליו', 'להחזר', 'היתה', 'מהנאמנות', 'בניית', 'בביתפקס', 'כוח',  
'וזאת', 'הדי', 'לעידוד', 'השקעה', 'מחול', 'שחולקו', 'מעל', 'בשווי',  
'העברתי', 'ימי', 'שחלה', 'השותפות', 'הבית', 'המיקוד', 'ומלאי', 'מסייע',  
'שפרטיו', 'מצויני', 'איש', 'הקשר', 'במס', 'ילדי', 'פרסו', 'וברחוב',  
'שידור', 'עזר', 'הנסחרי', 'ושווי', 'בהכנסות', 'חות', 'שניהלתי', 'ייפוי',  
'חובת', 'שלי', 'אינני', 'מתקיימת', 'פטורותחייבות', 'עיקרי', 'אלמ',  
'באנרגיות', 'רושמת', 'מניירות', 'לנקבה', 'אלמשרד', 'בסכו', 'פרויקט', 'ליב',  
'לחתו', 'מאנרגיה', 'אתר', 'ואני', 'שחל', 'נקטתי', 'כפולה', 'לתשומת',  
'מתחדשת', 'קייבו', 'עליה', 'מניות', 'לטלפו', 'מיקוד', 'חשמל', 'חלקי',  
'למרש', 'להעביר', 'העסקי', 'מהותי', 'חייבבקשה', 'האינטרנט', 'אחרי',  
'לאפשר', 'באמצעות', 'הנייד', 'מעסקממשלח', 'לרשות', 'הדואר', 'ממימוש',  
'מטבע', 'מעטי', 'עמי', 'וארבעה', 'ממוחשב', 'באינטראנט', 'לידיעת', 'לקייבו',  
'וירטואלי', 'המרה', 'חבר', 'מעוניי', 'בחברת', 'לכתובת', 'במקור', 'לעדכו',  
'התשסה', 'שמי', 'התשעז', 'ראשוני', 'הודעות', 'זיכויי', 'רלבנטיי', 'בעדכו',  
'למטבעות', 'מועברת', 'לתקנה', 'קבלת', 'אישי', 'נייד', 'המיסי', 'זוגו',  
'ניכוי', 'לחשבונני', 'לסעי', 'החזר', 'המתנהל', 'בבנק', 'מתאימה', 'חתימה',

'לצור', 'ושלושה', 'האלקטרוני', 'יועבר', 'שינוי', 'ששת', 'מגיע', 'רווק',  
'למשלוח', 'וחמישה', 'וא', 'שו', 'סהיכ', 'מייל', 'גרסה', 'גרוש', 'ארבעה',  
'זידני', 'החברה', 'שישה', 'מעמ', 'ושבעה', 'חדצידית', 'מסרו', 'וס', 'סט',  
'ותשעה', 'וט', 'רז', 'במרשמי', 'כפר', 'שמופיע', 'ושתיי', 'תז', 'שת',  
'שכירה', 'כפי', 'פתח', 'בנק', 'הדו', 'באני', 'עשר', 'מספרגרסה', 'סני',  
'ששי', 'סבא', 'וארבעי', 'שני', 'ליחיד', 'הגני', 'שי', 'בשוי', 'יפו',  
'בבשנת', 'שנתי', 'רזתושב', 'יצחק', 'בא', 'המ', 'חמישה', 'בס', 'גזש', 'מיל',  
'מסר', 'סא', 'יי', 'שוג', 'הדות', 'תקוה', 'אס', 'השרו', 'מחיסולית',  
'סלוניקי', 'ואט', 'מו', 'סכ', 'ותשעי', 'סו', 'קהילת', 'ושמוני', 'וסט',  
'רמת', 'הרצליה', 'ראה', 'וח', 'שחדצידית', 'ושבעי', 'דוד', 'נספחי',  
'מצבטופס', 'במשבצת', 'יעקב', 'ראש', 'בדות', 'יהודה', 'ועשר', 'בכ', 'רמלה',  
'בבבת', 'טס', 'סש', 'יוס', 'חולו', 'העי', 'המשרדד', 'גור', 'וי', 'כא',  
'צאט', 'וה', 'הי', 'אה', 'וששי', 'וג', 'בב', 'דות', 'עצמאי', 'אא', 'קיי',  
'הדוה', 'הא', 'הוגש', 'שבעה', 'ולא', 'הט', 'שוי', 'התשע', 'וש', 'סוט',  
'טוט', 'ני', 'זחדצידית', 'סס', 'זכ', 'חברות', 'שה', 'ות', 'סמ', 'מע', 'תד',  
'רעננה', 'טעי', 'שס', 'המט', 'סוס', 'נותני', 'ציו', 'דניאל', 'הד', 'ושישי',  
'פר', 'ידני', 'ועשרת', 'עפ', 'מקצועות', 'קהלת', 'סע', 'וזו', 'היא', 'גרשו',  
'מזהירה', 'מספ', 'הח', 'טופט', 'אונו', 'רו', 'הננ', 'השבו', 'דנה', 'סח',  
'אברה', 'ווק', 'ובחו', 'אלא', 'איל', 'תיז', 'אבי', 'תקווה', 'המתאימה',  
'מיקוה', 'מציב', 'בקשה', 'טו', 'פס', 'סה', 'שמואל', 'הצ', 'לאומי', 'מא',  
'דו', 'וטס', 'וה', 'החדה', 'משפחה', 'גהש', 'דתשתיפוס', 'אנ', 'מל',  
'עדי', 'דירה', 'נני', 'וק', 'שלמה', 'בבת', 'צבי', 'בח', 'חי', 'הכנסת',  
'חברת', 'הש', 'דוה', 'מט', 'תי', 'בלא', 'חל', 'הס', 'החסס', 'תשעה', 'בהול',  
'אלקטרוני', 'מעסקממשלת', 'ושישה', 'כת', 'ושלושת', 'שמ', 'הזו', 'וטא', 'ססט',  
'טסו', 'טוג', 'א', 'ובת', 'קיוסו', 'ב', 'בדיוות', 'מיכאל', 'שידני', 'ושני',  
'עה', 'ראשו', 'ברק', 'ממ', 'רויכמ', 'לוי', 'הוד', 'וחמשת', 'יב', 'יור',  
'למישלו', 'בתושב', 'בר', 'לסמ', 'בית', 'למשלוח', 'קוד', 'בעלי', 'כל',  
[ 'החדה', 'פרו', 'תייבת', 'דר

## ספר 3: בניית המודל של המכונה למודת

בספר זה נדבר יצרית המכונה הלומדת. בספר זה מנראה איך יצרנו את המכונה למודת איך אימנו אותה. ואיך מתבצאה תהליך החיזוי

### import

In [8]:

```
import tensorflow as tf # ספרייה ליצירת רשת נוירונים
import numpy as np # ספרייה לפעולות מתמטיות
import enum # ספריית החזרה להוספת
import config # דף הגדרות
training_size = 140 # כמות המסמכים שמוגדרים ללמידה
```

```
class Location(enum.IntEnum): # מקום של כל ערך ברשימה
    Lable = 0 # סוג הדוח
    Root = 1 # מיקום הקובץ
    wordtoken = 2 # המילים בקובץ
    bagofword = 3 # מבני הנתונים
    clientname = 4 # שם הלקוח
```

## פונקציה ראשונה בניית המודל

הפונקציה מקבלת שני מערכים. אחד מערך של אינפוט למכונה, והשני את הסוג של המסמך המתאים לכל אינפוט

הפונקציה בונה מכונה ומאמנת אותה לפי המערכים שהתקבלו

In [9]:

```
def buildmodel(labels, input):
    training_input = input[0:training_size]
    testing_input = input[training_size:]
    training_labels = labels[0:training_size]
    testing_labels = labels[training_size:]
    training_labels = np.array(training_labels).astype(np.uint8)
    training_padded = np.array(training_input)
    testing_labels = np.array(testing_labels).astype(np.uint8)
    testing_padded = np.array(testing_input)
    model = tf.keras.models.Sequential()
    model.add(tf.keras.layers.Flatten())
    model.add(tf.keras.layers.Dense(16, activation=config.ACTIVATION))
    model.add(tf.keras.layers.Dense(2, activation=config.ACTIVATION))
    model.compile(loss=config.LOSSFUNCTION, optimizer=config.OPTIMIZER,
metrics=[config.METRICS])
    model.fit(training_padded, training_labels, epochs=200,
validation_data=(testing_padded, testing_labels), verbose=2)
    model.save(config.MODELNAME)
את הפונקציה נחלק ל2
```

## חיזוי הנתונים

הפונקציה מקבל מערך אינפוט למערכת ואת המודל

הפונקציה מחשבת לכל מסמך את הסתברות הגובה ביותר לסוג בסמך ומחזירה את הכותרת המתאימה למסמך אם הוא עבר את ה-90%

In [34]:

```
def predict(model, data):
    for index, bagofword in enumerate(data):
        if len(bagofword) > 3: # בדיקה עם קיים אינפוט למערכת
            testing_padded =
np.array(bagofword[int(Location.bagofword)]).reshape(1, 900) # התאמת אינפוט
למוכנה

            predicted = model.predict(testing_padded)[0] # נסיון חיזוי
            print(predicted) # פקודה הצגה בשביל לראות את הערך קבנו
            label = np.argmax(predicted) # קחת את המספר הגבוה ביותר שקבנו
            if predicted[0] > 0.9 or predicted[1] > 0.9: # בדיקה אם אחד מין
הערכים עבר את ה-90%
                if label == 1: # בדיקה איזה ליבל יש לו את התוצאה הגודלה
ביותר
                    data[index][int(Location.Lable)] = config.REPORTTYPE2 #
הכנסת סוג מסמך למודל
                elif label == 0:
                    data[index][int(Location.Lable)] = config.REPORTTYPE1
    return data
```

## ריצת התוכנית לפי מה שניתן לראות כאן

כאן ניתן לראות בתוצאה שיש הצלחה של 97% לזיהוי המסמך. והמערכת הכינסה את הנתונים לתוך המבנה הנתונים

In [36]:

```
import warnings
import ParallelCode
import LoudModel
import os
warnings.filterwarnings("ignore")
model = tf.keras.models.load_model(config.MODELNAME)
data = ParallelCode.bagofwordonlytheard(os.path.join("example"))
predict(model, data)

WARNING:tensorflow:5 out of the last 5 calls to <function
Model.make_predict_function.<locals>.predict_function at
0x000002CC50F3D280> triggered tf.function retracing. Tracing is expensive
and the excessive number of tracings could be due to (1) creating
@tf.function repeatedly in a loop, (2) passing tensors with different
shapes, (3) passing Python objects instead of tensors. For (1), please
define your @tf.function outside of the loop. For (2), @tf.function has
experimental_relax_shapes=True option that relaxes argument shapes that can
avoid unnecessary retracing. For (3), please refer to
https://www.tensorflow.org/guide/function#controlling_retracing and
https://www.tensorflow.org/api_docs/python/tf/function for more details.
WARNING:tensorflow:5 out of the last 5 calls to <function
Model.make_predict_function.<locals>.predict_function at
0x000002CC50F3D280> triggered tf.function retracing. Tracing is expensive
and the excessive number of tracings could be due to (1) creating
@tf.function repeatedly in a loop, (2) passing tensors with different
shapes, (3) passing Python objects instead of tensors. For (1), please
define your @tf.function outside of the loop. For (2), @tf.function has
experimental_relax_shapes=True option that relaxes argument shapes that can
avoid unnecessary retracing. For (3), please refer to
```

[https://www.tensorflow.org/guide/function#controlling\\_retracing](https://www.tensorflow.org/guide/function#controlling_retracing) and  
[https://www.tensorflow.org/api\\_docs/python/tf/function](https://www.tensorflow.org/api_docs/python/tf/function) for more details.  
[0.9734049 0.01304069]

Out [36]:

[[example2015.pdf, 'דוחות שנתיים']]

טופס, אל, שלוש, מאות, ואחד, שודר, עי, מייצג, באינטרנט, 'אחד, מתו, ארבעה, התשמיס, ליחיד, וו, וו, אסמכתא, חותמת, המשרד, הרשו, בלשו, זכר, מתיחס, לנקבה, די, וחשבו, על, ההכנסות, באר, ובחול, תשע, תשע, תארי, הגשת, הדוח, ור, הדוח, בשנת, המס, אלפיי, ושמונהעשרה, שתיי, השנה, המתחילה, והמסתיימת, השומה, תל, אביב, חמש, חוליה, שבעה, עפי, סעי, מאה, ושלוש, ואחת, לפקודת, מס, הכנסה, רשאי, פקיד, השומה, לראות, מי, שהגיש, דוח, שלא, כללי, מולא, כראוי, או, שלא, צורפו, אליו, המסמכי, המתאימי, כמי, שלא, הגיש, דוח, על, הכנסות, והכנסות, בת, זוגי, שלושה, הכנסות, בלבד, חמש, אני, מגיש, דוח, לשנת, מס, זו, למרות, שאני, חייבבבקה, להחזיר, מס, וגי, מגיש, דוח, נפרד, מצב, הדוחהצהרה, של, בת, זוגי, אי, הכנסה, לב, בת, זוגי, בת, זוגי, עזר, לי, בהגשת, הכנסה, תארי, הגעה, זוג, רשו, סה, משות, לבני, הזוג, ללא, כ, עמדת, בתנאי, סעי, לפקודה, שלושה, לא, עמדת, בתנאי, סעי, לפקודה, עולה, חדש, תושב, חוזר, ותיק, רתושב, חוזר, וחלות, על, הכנסות, מחול, הקלות, במס, ההדהדק, לקדה, קמס, היו, לי, או, לב, בת, זוגי, או, לילדי, שטר, מלאו, לה, נכסי, בחול, בשווי, של, שח, או, יותר, המס, היתה, לי, או, לב, בת, זוגי, הכנסה, חייבת, כהגדרתה, בסעי, בה, לפקודה, העולה, על, שח, מס, היה, לי, לב, בת, זוגי, מחזור, מכירות, מניירות, ער, הנסחר, בבורסה, שאינו, פטור, ממס, העולה, על, שח, תארי, הגעה, בת, הזוג, המס, היו, לי, הכנסות, מפעילות, באינטרנט, מסחר, שיווק, פרסו, וכד, בבשנת, המס, היו, לי, הכנסות, בהתא, לחוק, אנרגיות, מתחדשות, המס, היו, לי, הכנסות, ממימוש, מטבע, וירטואלי, לרבות, המרה, למטבעות, אחרי, בני, הזוג, עיוור, או, נכה, לפי, סעי, לפקודה, בשנת, המס, העברתי, במש, שתיי, עשרה, חדשי, כספי, אל, מחו, לישראל, בסכו, כולל, של, שח, או, יותר, ספח, לחישוב, ההכנסה, בגי, תשלומי, עודפי, של, מעביד, לקר, השתלמות, וקופג, טופס, שב, ישראל, מתקימת, לגבי, חזקת, ימי, שהיה, בישראל, הנסחרת, על, ידי, ואני, חייב, בהגשת, דוח, לפי, סעי, לפקודה, מצב, טופס, בנאמנות, ברשומה, וברחיוב, דוח, זה, כולל, את, הכנסות, ואת, הכנסות, הנאמנות, מצב, טופס, נה, בנאמנות, שחלה, עליו, חובת, דיווח, לפי, סעי, אב, לפקודה, תושב, ישראל, שמלאו, לו, עשרי, וחמש, שנה, ושווי, נכסי, הנאמנות, עולה, על, שח, נה, בנאמנות, שההכנסות, שחולקו, לי, מהנאמנות, כלולות, בדוח, זה, מצב, העתק, טופס, נה, בנאמנות, שממנה, היו, לי, חלוקות, פטורותחייבות, בשנת, המס, כהגדרת, בסעי, הרשומות, בדוח, זה, בשדה, מאתי, ושבעי, ואחת, וזיק, זכאי, בשותפות, נפט, ביו, ודיווחתי, על, חלקי, בהכנסות, השותפות, זוגי, חבר, קיבו, מס, תיק, והכנסה, חייבת, מועברת, לקיבו, בהתא, לסעי, לפקודה, הזוג, הרשו, הזוג, גל, מניות, מהותי, בחברת, מעטי, שחל, עליה, סעי, לפקודה, כיר, עמי, ודיווחתי, בהתא, לתקנה, לפי, סעי, לפקודה, הכנסות, מבני, לפי, סעי, אג, לפקודה, הדות, כולל, דיווח, על, סיו, בניית, פרויקט, שתיי, מצב, טופס, שבע, מאות, ושניי, המחזור, מכלל, שליטה, בחבר, בני, אד, תושב, חו, נסחר, בחול, מצב, טופס, מאה, וחמישי, לא, העסקי, שלי, או, של, זכויות, בחבר, בני, אד, תושב, חו, שאינו, נסחר, מצב, טופס, מאה, וחמישי, לא, בת, זוגי, הוא, מעל, אפס, היו, לי, או, לב, בת, זוגי, עסקאות, צדדי, קשורי, בחול, כמשמעות, בסעי, לפקודה, מצב, טופס, אל, שמונה, מאות, ושמוני, וחמש, לא, אפס, שיח, ולה, החייבת, בדיווח, מכוה, סעי, לפקודה, שתיי, מצב, טופס, אל, מאתי, ושלוש, עשרה, לא, לא, מעמ, ווות,

```
array([1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1,
       1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1,
       1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1])
```



```
1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 0,
0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0,
1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0,
1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1,
1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1,
1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1,
1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1,
1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1,
0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0,
1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0,
0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1,
1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0,
1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ]]
```

In [ ]:

## Object Detection API Demo



[Run in Google Colab](#)



[View source on GitHub](#)

Welcome to the [Object Detection API](#). This notebook will walk you step by step through the process of using a pre-trained model to detect objects in an image.

**Important:** This tutorial is to help you through the first step towards using [Object Detection API](#) to build models. If you just need an

off the shelf model that does the job, see the [TFHub object detection example](#).

## Setup

Important: If you're running on a local machine, be sure to follow the [installation instructions](#). This notebook includes only what's necessary to run in Colab.

## Install

In [1]:

```
#!pip install -U --pre tensorflow=="2.*"
!pip install tf_slim
Defaulting to user installation because normal site-packages is not
writable
Collecting tf_slim
  Downloading tf_slim-1.1.0-py2.py3-none-any.whl (352 kB)
Requirement already satisfied: absl-py>=0.2.2 in
c:\users\user\appdata\roaming\python\python38\site-packages (from tf_slim)
(0.11.0)
```

```
Requirement already satisfied: six in
c:\users\user\appdata\roaming\python\python38\site-packages (from abs-
py>=0.2.2->tf_slim) (1.15.0)
Installing collected packages: tf-slim
Successfully installed tf-slim-1.1.0
Make sure you have pycocotools installed
```

In [2]:

```
!pip install pycocotools
Defaulting to user installation because normal site-packages is not
writeable
Collecting pycocotools
  Downloading pycocotools-2.0.2.tar.gz (23 kB)
Requirement already satisfied: setuptools>=18.0 in c:\program
files\python38\lib\site-packages (from pycocotools) (49.2.1)
Collecting cython>=0.27.3
  Using cached Cython-0.29.21-cp38-cp38-win_amd64.whl (1.7 MB)
Collecting matplotlib>=2.1.0
  Downloading matplotlib-3.3.3-cp38-cp38-win_amd64.whl (8.5 MB)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.3 in
c:\users\user\appdata\roaming\python\python38\site-packages (from
matplotlib>=2.1.0->pycocotools) (2.4.7)
Requirement already satisfied: pillow>=6.2.0 in
c:\users\user\appdata\roaming\python\python38\site-packages (from
matplotlib>=2.1.0->pycocotools) (8.1.0)
Collecting cyclor>=0.10
  Downloading cyclor-0.10.0-py2.py3-none-any.whl (6.5 kB)
Requirement already satisfied: python-dateutil>=2.1 in
c:\users\user\appdata\roaming\python\python38\site-packages (from
matplotlib>=2.1.0->pycocotools) (2.8.1)
Requirement already satisfied: numpy>=1.15 in
c:\users\user\appdata\roaming\python\python38\site-packages (from
matplotlib>=2.1.0->pycocotools) (1.19.5)
Collecting kiwisolver>=1.0.1
  Downloading kiwisolver-1.3.1-cp38-cp38-win_amd64.whl (51 kB)
Requirement already satisfied: six in
c:\users\user\appdata\roaming\python\python38\site-packages (from
cyclor>=0.10->matplotlib>=2.1.0->pycocotools) (1.15.0)
Building wheels for collected packages: pycocotools
  Building wheel for pycocotools (setup.py): started
  Building wheel for pycocotools (setup.py): finished with status 'done'
  Created wheel for pycocotools: filename=pycocotools-2.0.2-cp38-cp38-
win_amd64.whl size=81474
sha256=fe14330cccd9a9a77ab7ed38f8a298334aaafe57ba93beb87d0cdc69d95209b0
  Stored in directory:
c:\users\user\appdata\local\pip\cache\wheels\e7\77\b2\6f38b5bea571cd8f4689f
91a7c1ed2eaecb2c2ce17f9945b17
Successfully built pycocotools
Installing collected packages: kiwisolver, cyclor, matplotlib, cython,
pycocotools
Successfully installed cyclor-0.10.0 cython-0.29.21 kiwisolver-1.3.1
matplotlib-3.3.3 pycocotools-2.0.2
Get tensorflow/models or cd to parent directory of the repository.
```

In [ ]:

```
import os
import pathlib

if "models" in pathlib.Path.cwd().parts:
    while "models" in pathlib.Path.cwd().parts:
        os.chdir('..')
elif not pathlib.Path('models').exists():
```

```
!git clone --depth 1 https://github.com/tensorflow/models
Compile protobufs and install the object_detection package
```

In [ ]:

```
%%bash
cd models/research/
protoc object_detection/protos/*.proto --python_out=.
```

In [ ]:

```
%%bash
cd models/research
pip install .
```

## Imports

In [1]:

```
import numpy as np
import os
import six.moves.urllib as urllib
import sys
import tarfile
import tensorflow as tf
import zipfile

from collections import defaultdict
from io import StringIO
from matplotlib import pyplot as plt
from PIL import Image
from IPython.display import display
Import the object detection module.
```

In [4]:

```
from object_detection.utils import ops as utils_ops
from object_detection.utils import label_map_util
from object_detection.utils import visualization_utils as vis_util
Patches:
```

In [5]:

```
# patch tf1 into `utils.ops`
utils_ops.tf = tf.compat.v1

# Patch the location of gfile
tf.gfile = tf.io.gfile
```

## Model preparation

### Variables

Any model exported using the `export_inference_graph.py` tool can be loaded here simply by changing the path.

By default we use an "SSD with Mobilenet" model here. See the [detection model zoo](#) for a list of other models that can be run out-of-the-box with varying speeds and accuracies.

### Loader

In [6]:

```
def load_model(model_name):
    base_url = 'http://download.tensorflow.org/models/object_detection/'
    model_file = model_name + '.tar.gz'
    model_dir = tf.keras.utils.get_file(
        fname=model_name,
        origin=base_url + model_file,
        untar=True)

    model_dir = pathlib.Path(model_dir)/"saved_model"
```

```
model = tf.saved_model.load(str(model_dir))
```

```
return model
```

## Loading label map

Label maps map indices to category names, so that when our convolution network predicts 5, we know that this corresponds to airplane. Here we use internal utility functions, but anything that returns a dictionary mapping integers to appropriate string labels would be fine

In [7]:

```
# List of the strings that is used to add correct label for each box.
PATH_TO_LABELS = 'object_detection\images\labelmap.pbtxt'
category_index =
label_map_util.create_category_index_from_labelmap(PATH_TO_LABELS,
use_display_name=True)
```

For the sake of simplicity we will test on 2 images:

In [2]:

```
import pathlib
# If you want to test the code with your images, just add path to the
images to the TEST_IMAGE_PATHS.
PATH_TO_TEST_IMAGES_DIR = pathlib.Path('object_detection/test_images')
TEST_IMAGE_PATHS = sorted(list(PATH_TO_TEST_IMAGES_DIR.glob("*.jpeg")))
TEST_IMAGE_PATHS
```

Out[2]:

```
[WindowsPath('object_detection/test_images/data (1).jpeg'),
 WindowsPath('object_detection/test_images/data (10).jpeg')]
```

## Detection

Load an object detection model:

In [9]:

```
# model_name = 'ssd_mobilenet_v1_coco_2017_11_17'
# detection_model = load_model(model_name)
detection_model =
tf.saved_model.load('object_detection\inference_graph\saved_model')
```

In [10]:

```
print(detection_model.signatures['serving_default'].inputs)
[<tf.Tensor 'input_tensor:0' shape=(1, None, None, 3) dtype=uint8>,
<tf.Tensor 'unknown:0' shape=<unknown> dtype=resource>, <tf.Tensor
'unknown_0:0' shape=<unknown> dtype=resource>, <tf.Tensor 'unknown_1:0'
shape=<unknown> dtype=resource>, <tf.Tensor 'unknown_2:0' shape=<unknown>
dtype=resource>, <tf.Tensor 'unknown_3:0' shape=<unknown> dtype=resource>,
<tf.Tensor 'unknown_4:0' shape=<unknown> dtype=resource>, <tf.Tensor
'unknown_5:0' shape=<unknown> dtype=resource>, <tf.Tensor 'unknown_6:0'
shape=<unknown> dtype=resource>, <tf.Tensor 'unknown_7:0' shape=<unknown>
dtype=resource>, <tf.Tensor 'unknown_8:0' shape=<unknown> dtype=resource>,
<tf.Tensor 'unknown_9:0' shape=<unknown> dtype=resource>, <tf.Tensor
'unknown_10:0' shape=<unknown> dtype=resource>, <tf.Tensor 'unknown_11:0'
shape=<unknown> dtype=resource>, <tf.Tensor 'unknown_12:0' shape=<unknown>
dtype=resource>, <tf.Tensor 'unknown_13:0' shape=<unknown> dtype=resource>,
dtype=resource>, <tf.Tensor 'unknown_103:0' shape=<unknown>
dtype=resource>, <tf.Tensor 'unknown_104:0' shape=<unknown>
```

And returns several outputs:

In [11]:

```
detection_model.signatures['serving_default'].output_dtypes
```

Out[11]:

```
{'detection_anchor_indices': tf.float32,
'detection_boxes': tf.float32,
'detection_classes': tf.float32,
```

```
'detection_multiclass_scores': tf.float32,
'detection_scores': tf.float32,
'num_detections': tf.float32,
'raw_detection_boxes': tf.float32,
'raw_detection_scores': tf.float32}
```

In [12]:

```
detection_model.signatures['serving_default'].output_shapes
```

Out[12]:

```
{'detection_anchor_indices': TensorShape([1, 100]),
'detection_boxes': TensorShape([1, 100, 4]),
'detection_classes': TensorShape([1, 100]),
'detection_multiclass_scores': TensorShape([1, 100, 1]),
'detection_scores': TensorShape([1, 100]),
'num_detections': TensorShape([1]),
'raw_detection_boxes': TensorShape([1, 49104, 4]),
'raw_detection_scores': TensorShape([1, 49104, 1])}
```

Add a wrapper function to call the model, and cleanup the outputs:

In [13]:

```
def run_inference_for_single_image(model, image):
    image = np.asarray(image)
    # The input needs to be a tensor, convert it using
    `tf.convert_to_tensor`.
    input_tensor = tf.convert_to_tensor(image)
    # The model expects a batch of images, so add an axis with `tf.newaxis`.
    input_tensor = input_tensor[tf.newaxis,...]

    # Run inference
    model_fn = model.signatures['serving_default']
    output_dict = model_fn(input_tensor)

    # All outputs are batches tensors.
    # Convert to numpy arrays, and take index [0] to remove the batch
    dimension.
    # We're only interested in the first num_detections.
    num_detections = int(output_dict.pop('num_detections'))
    output_dict = {key:value[0, :num_detections].numpy()
                    for key,value in output_dict.items()}
    output_dict['num_detections'] = num_detections

    # detection_classes should be ints.
    output_dict['detection_classes'] =
output_dict['detection_classes'].astype(np.int64)

    # Handle models with masks:
    if 'detection_masks' in output_dict:
        # Reframe the the bbox mask to the image size.
        detection_masks_reframed = utils_ops.reframe_box_masks_to_image_masks(
            output_dict['detection_masks'],
output_dict['detection_boxes'],
            image.shape[0], image.shape[1])
        detection_masks_reframed = tf.cast(detection_masks_reframed > 0.5,
            tf.uint8)
        output_dict['detection_masks_reframed'] =
detection_masks_reframed.numpy()
    return output_dict
```

Add a crop function to call the model, and cleanup the outputs:

In [56]:

```
def crop_select_only(image_np,output_dict):
    img_height, img_width, img_channel = image_np.shape
    absolute_coord = []
```

```
N = len(output_dict['detection_boxes'])
for i in range(N):
    if output_dict['detection_scores'][i] > 0.9:
        box = output_dict['detection_boxes'][i]
        ymin, xmin, ymax, xmax = box
        x_up = int(xmin*img_width)
        y_up = int(ymin*img_height)
        x_down = int(xmax*img_width)
        y_down = int(ymax*img_height)
        absolute_coord.append((x_up,y_up,x_down,y_down))

bounding_box_img = []
for c in absolute_coord:
    bounding_box_img.append(image_np[c[1]:c[3], c[0]:c[2],:])
for image in bounding_box_img:
    display(Image.fromarray(image))
```

Run it on each test image and show the results:

In [59]:

```
def show_inference(model, image_path):
    # the array based representation of the image will be used later in order
    # to prepare the
    # result image with boxes and labels on it.
    image_np = np.array(Image.open(image_path))
    # Actual detection.
    output_dict = run_inference_for_single_image(model, image_np)
    # Visualization of the results of a detection.
    vis_util.visualize_boxes_and_labels_on_image_array(
        image_np,
        output_dict['detection_boxes'],
        output_dict['detection_classes'],
        output_dict['detection_scores'],
        category_index,
        instance_masks=output_dict.get('detection_masks_reframed', None),
        use_normalized_coordinates=True,
        line_thickness=8)
    crop_select_only(image_np,output_dict)
    # display(Image.fromarray(image_np))

for image_path in TEST_IMAGE_PATHS:
    show_inference(detection_model, image_path)
```

In [60]:

## Instance Segmentation

In [20]:

```
model_name = "mask_rcnn_inception_resnet_v2_atrous_coco_2018_01_28"
masking_model = load_model(model_name)
Downloading data from
http://download.tensorflow.org/models/object_detection/mask_rcnn_inception_
resnet_v2_atrous_coco_2018_01_28.tar.gz
93536256/727390102 [==>.....] - ETA: 1:20
-----
KeyboardInterrupt                                Traceback (most recent call last)
<ipython-input-20-c5ec5a681b2e> in <module>
      1 model_name = "mask_rcnn_inception_resnet_v2_atrous_coco_2018_01_28"
----> 2 masking_model = load_model(model_name)

<ipython-input-4-ac8277046829> in load_model(model_name)
```

```

2     base_url =
'http://download.tensorflow.org/models/object_detection/'
3     model_file = model_name + '.tar.gz'
----> 4     model_dir = tf.keras.utils.get_file(
5         fname=model_name,
6         origin=base_url + model_file,

~\AppData\Roaming\Python\Python38\site-
packages\tensorflow\python\keras\utils\data_utils.py in get_file(fname,
origin, untar, md5_hash, file_hash, cache_subdir, hash_algorithm, extract,
archive_format, cache_dir)
273         try:
274             try:
--> 275                 urlretrieve(origin, fpath, dl_progress)
276             except HTTPError as e:
277                 raise Exception(error_msg.format(origin, e.code, e.msg))

C:\ProgramData\Anaconda3\lib\urllib\request.py in urlretrieve(url,
filename, reporthook, data)
274
275         while True:
--> 276             block = fp.read(bs)
277             if not block:
278                 break

C:\ProgramData\Anaconda3\lib\http\client.py in read(self, amt)
456             # Amount is given, implement using readinto
457             b = bytearray(amt)
--> 458             n = self.readinto(b)
459             return memoryview(b)[:n].tobytes()
460         else:

C:\ProgramData\Anaconda3\lib\http\client.py in readinto(self, b)
500             # connection, and the user is reading more bytes than will
be provided
501             # (for example, reading in 1k chunks)
--> 502             n = self.fp.readinto(b)
503             if not n and b:
504                 # Ideally, we would raise IncompleteRead if the
content-length

C:\ProgramData\Anaconda3\lib\socket.py in readinto(self, b)
667         while True:
668             try:
--> 669                 return self._sock.recv_into(b)
670             except timeout:
671                 self._timeout_occurred = True

```

### KeyboardInterrupt:

The instance segmentation model includes a `detection_masks` output:

```

masking_model.output_shapes                                     In [ ]:

for image_path in TEST_IMAGE_PATHS:                           In [ ]:
    show_inference(masking_model, image_path)

```

## Book 5: Building the AI API

### api של ב class בספר זה מדבר על בניית ה

ai-זה הוא ספר האחרון אשר מחבר את כל התהליכים בספרים הקודמים לתהליך אחד אשר מרכיב את ה

import

In [2]:

```
import os # צויין קודם
import ParallelCode #2 ספר
import MachineLearning #3 ספר
import random # ספרייה הרחבה לפעולות עקריות
import pickle # צויין קודם
import enum # צויין קודם
import numpy as np # צויין קודם
import tensorflow as tf # צויין קודם
import GetNamesByClass #4 ספר
import csv # ספריית הרחבה לפעולות על csv
import config # צויין קודם
import HebrewNlp #1 ספר
import CleanText # 1 ספר
```

```
class Location(enum.IntEnum):
    Lable = 0 # סוג הדוח
    Root = 1 # מיקום הקובץ
    wordtoken = 2 # המילים בקובץ
    bagofword = 3 # מבני הנתונים
    clientname = 4 # שם הלקוח
```

### model traing class

פה בעצם יצירת המודל מתקיימת api של ה class זה

In [3]:

```
class ModelTrainig:
    def __init__(self,): # class -אובייקטים של ה
        self.bagofwords = {} # של כל המילים בעולם שלנו template
        self.trainiset= [] # מידע ללאימון המוכנה
        self.category_index # אינדקס המידע לזיהוי נתונים
        self.detection_model # מוכנה של ספר 4
        self.allfirstname # כל השמות הפרטים העיברים
        self.alllastname# כל השמות המשפחה העיברים
```

### class function

#### פונקציות לטעינת המוכנה

template-הפונקציה הזאת טוענת את ה

ומאגר המילים לתוך האובייקטים של הקלאס

In [5]:

```
def Loadbegofworld(self):
    if (not
os.path.exists(os.path.join(config.WORKSPACE,config.SUFFLETRAINFILE))): #
    בדיקה אם הקבצים שראינו בספר 2 קיימים
        (self.bagofwords, self.trainiset) = ParallelCode.begofword() # אם
    וליצור קבצים מתאימים data set ללא קיימים אז להפעיל את התהליך שבספר 2 על כל ה
```



```
random.shuffle(self.trainiset) # עירוב הדאטה סט בשביל לעלות את
האמינות המכונה
file = open(os.path.join(config.WORKSPACE,
config.SUFFLETRAINFILE), 'wb') # ירצת קובץ חדש של הדאטה מעורבב
pickle.dump(self.trainiset, file)
file.close()
else:
file = open(os.path.join(config.WORKSPACE,
config.BAGOFWORDSFILE), 'rb') # הכנסת הנתונים המתאים מהקבצים הנצרו בספר 2
self.bagofwords = pickle.load(file)
file.close()
file = open(os.path.join(config.WORKSPACE,
config.SUFFLETRAINFILE), 'rb') # הכנסת הנתונים המתאים מהקבצים הנצרו בספר 2
self.trainiset = pickle.load(file)
file.close()
```

### בניית המודל של ספר 3

הפונקציה מכינה מחלק את הדאטה לשני המערכים במתאים לספר 3. ומכינה את המוכנה

In [ ]:

```
def buildthemodel(self):
    labels=[] # מערך תוויות
    files = [] # מערך האינפוט
    for item in self.trainiset: # יצירת מערך לשליחה לספר 3
        npitem = np.array(item[int(Location.bagofword)])
        files.append(item[int(Location.bagofword)])
        if item[int(Location.Lable)] == config.REPORTTYPE2: # בדיקת
            התווית והכנסה לתוך מערך התווים
            labels.append(1)
        else:
            labels.append(0)
    MachineLearning.buildmodel(labels,files) # פונקציה הכנת במודל
    מספר 3
```

### בניית המודל של ספר 3-4

הפונקציה הזאת היא טוענת את המודל של ספר 4 וספר 3

In [8]:

```
def LoadModel(self):
    try:
        self.model = tf.keras.models.load_model(config.MODELNAME) #
        נסיון טעינה של ספר 3 אם קיים
    except Exception as e:
        self.model = self.buildthemodel() # יצירת מודל של ספר 3 עם לא
        קיים
    self.category_index, self.detection_model =
    GetNamesByClass.Load_Pic_Model() # טעינה ספר 4
```

### טעינת הקבצים של

In [9]:

```
def LoadNames(self):
    allname = [] # מערך שמות פרטים
    with open(config.FIRSTNAME, encoding=config.ENCODING, newline='')
    as csvfile: # טעינה מקובץ csv
        reader = csv.reader(csvfile, delimiter='\\t')
        for row in reader:
            data = ', '.join(row)
```

```

        allname.append(data)
    self.allfirstname = allname
    allname = [] # מערך שמות המשפחה
    with open(config.LASTNAME, encoding=config.ENCODING, newline='') as
csvfile: # טעינה מקובץ csv
        reader = csv.reader(csvfile, delimiter='\\t')
        for row in reader:
            data = ', '.join(row)
            allname.append(data)
    self.alllastname = allname

```

## טעינת הפרויקט

הפונקציה הזאת טוענת את כל המכונות ואת כל הקבצים לפרויקט

In [10]:

```

def LoadTheApi(self):
    self.Loadbegofworld()
    self.LoadModel()
    self.LoadNames()
    try: # החיצונים תקנים api בדיקת אם ה
        HebrewNlp.HebrewTokenizer(config.ERRORLABEL)
        CleanText.Translate2Word("english")
    except Exception as e:
        raise e

```

## מיצאת השם הקובץ

הפונקציה הזאת מקבלת כתובת לקובץ שם קובץ הפקציה מוציאה את השם מהקובץ לפי ספר 4

In [11]:

```

def NameWithCat(self, file, name, root):
    append_lst = file.copy()
    append_lst.append(
        GetNamesByClass.Classtfy( # קריאה לספר
            self.category_index,
            self.detection_model,
            root, name,
            self.allfirstname,
            self.alllastname))
    return append_lst

```

## חזוי מסמך חדש

הפונקציה מקבלת תיקיה למיון המסמכים ופונקציה אומרת לען כל מסמך צריך להגיע

In [13]:

```

def predictAndClusterfy(self, when):
    data = ParallelCode.bagofwordonlytheard(when) # הכנת האינפוט למערכת
    ספר 1-2
    data_after_subject = MachineLearning.predict(self.model, data) #
    חזוי ספר 3
    allthesetup = []
    for file in data_after_subject: # יצירת המידע לשליחה לסרבר-לקוח
        if file[int(Location.Lable)] in config.REPORTTYPE1: # הכנסה
            תווים מתאימה לסוג מסמך והוצאת המידע לפי סוג מסמך
            root = file[int(Location.Root)]
            allname = root.split('\\\\')
            name = allname[(len(allname))-1]
            append_lst = self.NameWithCat(file, name, root) # שליחה
            לפונקציה הוצאת השם
            allthesetup.append(append_lst) # הופסת השם לרישמה
        else:

```

```

        append_lst = file.copy()

append_lst.append(HebrewNlp.GetName(file[int(Location.wordtoken)],self.allfirstname,self.alllastname))

allthesetup.append(append_lst) #1 הוצאת שם לפי ספר
return allthesetup

```

## יחד AI-תוצאה של כל ה

In [15]:

[illegible]

84 סיווג מסמכים אוטומטי | לב יודצ'ב \ עמרי גיגי

85 סיווג מסמכים אוטומטי | לב יודצ'ב \ עמרי גיגי

86 סיווג מסמכים אוטומטי | לב יודצ'ב \ עמרי גיגי

87 סיווג מסמכים אוטומטי | לב יודצ'ב \ עמרי גיגי

לפי, 'סעי', 'לפקודה', 'מצב', 'טופס', 'בנאמנות', 'ברשומה', 'וברחיוב',  
 'דוח', 'זה', 'כולל', 'את', 'הכנסות', 'ואת', 'הכנסות', 'הנאמנות', 'מצב',  
 'טופס', 'בנאמנות', 'שחלה', 'עליו', 'חובת', 'דיווח', 'לפי', 'סעי', 'אב',  
 'לפקודה', 'תושב', 'ישראל', 'שמלאו', 'לו', 'עשרי', 'וחמש', 'שנה', 'ושוו',  
 'בנאמנות', 'שההכנסות', 'שחולקו', 'לי', 'מהנאמנות', 'כלולות', 'בדוח', 'זה',  
 'מצב', 'העתק', 'טופס', 'אי', 'הכנסה', 'לב', 'בת', 'זוג', 'בת', 'זוגי',  
 'עזר', 'לי', 'בהשגת', 'הכנסה', 'בתנאי', 'סעי', 'לפקודה', 'לא', 'עמדת',  
 'בתנאי', 'סעי', 'לפקודה', 'מחול', 'הקלות', 'במס', 'אחד', 'מתו', 'ארבעה',  
 'ער', 'דח', 'שנתי', 'אלפי', 'ושש', 'עשרה', 'חותמת', 'המשרד', 'אפס',  
 'תארי', 'הגעה', 'זוג', 'רשו', 'אפס', 'תארי', 'הגעה', 'בת', 'הזוג', 'בבשנת',  
 'המס', 'היו', 'לי', 'הכנסות', 'בהתא', 'לחוק', 'אנרגיות', 'מתחדשות',  
 'ישראל', 'בסכו', 'כולל', 'של', 'שח', 'או', 'יותר', 'בנאמנות', 'שממנה',  
 'היו', 'לי', 'חלוקות', 'פטורות', 'בשנת', 'המס', 'כהגדרת', 'בסעי',  
 'הרשומות', 'בדוח', 'זה', 'ויק', 'זכאי', 'בשותפות', 'נפט', 'ביו',  
 'ודיווחתי', 'על', 'חלקי', 'בהכנסות', 'השותפות', 'זוגי', 'חבר', 'קיבו',  
 'מס', 'תיק', 'מניות', 'מהות', 'בחברת', 'מעטי', 'שחל', 'עליה', 'סעי',  
 'לפקודה', 'יר', 'עמי', 'ודיווחתי', 'בהתא', 'לתקנה', 'לפי', 'סעי', 'לפקודת',  
 'הכנסות', 'מבני', 'לפי', 'סעי', 'אג', 'לפקודה', 'הדוח', 'כולל', 'דיזוח',  
 'על', 'סיו', 'בניית', 'פרויקט', 'שליטה', 'בחבר', 'בני', 'אד', 'תושב', 'חו',  
 'נסחר', 'בחול', 'וכויות', 'בחבר', 'בני', 'אד', 'תושב', 'חו', 'שאינו',  
 'נסתר', 'אפס', 'היו', 'לי', 'או', 'לב', 'בת', 'זוגי', 'עסקאות', 'צדדי',  
 'קשורי', 'לה', 'החייבת', 'בדיווח', 'מכוח', 'סעי', 'לפקודה', 'ות', 'דעת',  
 'חייבת', 'בדיווח', 'המאפשרת', 'יתרו', 'מס', 'כאמור', 'בסעי', 'לפקודה',  
 'מדה', 'חייבת', 'בדיווח', 'הכלולה', 'ברשימה', 'שפרסמה', 'רשות', 'המס',  
 'כאמור', 'בסעי', 'לפקודה', 'משק', 'חקלאי', 'לי', 'שטח', 'אדמה', 'מעובד',  
 'מצב', 'טופס', 'אני', 'או', 'בת', 'זוגי', 'חשבונות', 'שניהלתי', 'עפי',  
 'בחול', 'כמשמעות', 'בסעי', 'לפקודה', 'הכנסה', 'חייבת', 'מועברת', 'לקיבו',  
 'בהתא', 'לסעי', 'נכסי', 'הנאמנות', 'עולה', 'על', 'שטח', 'בשדה', 'מאתי',  
 'ושבעי', 'ואחת', 'לפקודה', 'הזוג', 'הרשו', 'מצב', 'טופס', 'שבע',  
 'מאות', 'ושנתי', 'המחזור', 'מכלל', 'מצב', 'טופס', 'מאה', 'וחמישי', 'לא',  
 'העסקי', 'שלי', 'או', 'של', 'מצב', 'טופס', 'מאה', 'וחמישי', 'לא', 'בת',  
 'זוגי', 'אפס', 'מעל', 'מצב', 'טופס', 'אל', 'שלוש', 'מאות', 'ושמוני',  
 'וחמש', 'לא', 'אפס', 'מצב', 'טופס', 'אל', 'מאתי', 'ושלוש', 'עשרה', 'מצב',  
 'טופס', 'אל', 'שמונה', 'מאות', 'וארבעי', 'וחמש', 'לא', 'תשע', 'שודר',  
 'טופס', 'שישה', 'אלפי', 'מאה', 'ואחת', 'עשרה', 'חייב', 'מצב', 'טופס', 'אל',  
 'שלוש', 'מאות', 'וארבעי', 'ושש', 'שותפי', 'בשותפות', 'מצב', 'טופס',  
 'תוספת', 'סעי', 'להוראות', 'ניהול', 'ספרי', 'נסות', 'מעסקממשלח', 'יד',  
 'עיקרי', 'הדוח', 'מבוסס', 'על', 'פנקסי', 'עוסק', 'פטור', 'ניהלתי', 'הנהלת',  
 'חשבונות', 'כפולה', 'זחדצידית', 'הפעלתי', 'קופה', 'רושמת', 'זלא', 'הפקת',  
 'תיעוד', 'פני', 'זידני', 'ממוחשב', 'כלת', 'ניכוי', 'זיכוי', 'ניכוי',  
 'מס', 'במקור', 'הטבות', 'מס', 'יש', 'לצר', 'מסמכי', 'רלבנטי', 'אישי',  
 'הזוג', 'הרשו', 'בת', 'הזוג', 'מספר', 'תיק', 'לרבות', 'מי', 'שאינו',  
 'נשוי', 'או', 'וא', 'פרוד', 'הת', 'פחתי', 'בשנת', 'המס', 'נשוי', 'דרסה',  
 'תמיר', 'דרסה', 'קר', 'שלושה', 'גרשו', 'משפחה', 'פרטי', 'משפחה', 'פרטי',  
 'יעקב', 'ד', 'בהתא', 'למרש', 'רשות', 'המס', 'האב', 'תארי', 'לידה', 'האב',  
 'תארי', 'לידה', 'נולל', 'מיקוד', 'שלוסקי', 'עשרי', 'ושש', 'ראשו', 'לציו',  
 'שבעה', 'מיליו', 'חמש', 'מאות', 'שבעי', 'ושבעה', 'אל', 'שבע', 'מאות',  
 'ותשעעשרה', 'כתובת', 'למישלו', 'דואר', 'רכתוצת', 'המגורי', 'לטו', 'אפס',  
 'משי', 'שבעה', 'שתי', 'עשרה', 'חמש', 'לפ', 'היד', 'אק', 'הי', 'בביתפקס',  
 'בעבודה', 'הזוג', 'הרשו', 'בעבודה', 'בת', 'הזוג', 'טלפו', 'נייד',  
 'העיקרי', 'פרט', 'העסק', 'זספר', 'הבית', 'הישוב', 'המיקוד', 'אחד', 'זיק',  
 'בעסק', 'העיקרי', 'שבעי', 'אחד', 'מספ', 'תיק', 'ניכוי', 'מספר', 'עוסק',  
 'במעמ', 'שבעה', 'מספר', 'תיק', 'ניי', 'זב', 'מספר', 'עוסק', 'במעמ',  
 'מעבידי', 'אפס', 'שבעה', 'אפס', 'ברק', 'שבעה', 'מאתי', 'ושבעי', 'ושבעי',  
 'עשרי', 'ושמונה', 'דוסה', 'תמיר', 'אפס', 'המס', 'מגיע', 'יועבר', 'לחשבונני',  
 'המתנהל', 'על', 'שמי', 'בבנק', 'ראשוני', 'ואו', 'שינוי', 'פרטי', 'חשבו',  
 'הבנק', 'יש', 'לצר', 'אסמכתא', 'מתאימה', 'ריני', 'מצהיר', 'בזה', 'כי',  
 'בשנת', 'המס', 'לא', 'היו', 'לי', 'ולב', 'בת', 'זוגי', 'הכנסות', 'נוספות',  
 'על', 'אלו', 'הכלולות', 'בדוח', 'וכ', 'כי', 'הפרטי', 'שבדי', 'זחשבו', 'זה',  
 'ונספחיו', 'נכוני', 'ומלאי', 'ידוע', 'לי', 'שא', 'המסמכי', 'הוגשו', 'אופ',  
 'מקוו', 'עלי', 'לשמור', 'את', 'המסמכי', 'שצורפו', 'ואת', 'הקבלות',



89 סיווג מסמכים אוטומטי | לב יודצ'ב \ עמרי גיגי

90 סיווג מסמכים אוטומטי | לב יודצ'ב \ עמרי גיגי

91 סיווג מסמכים אוטומטי | לב יודצ'ב \ עמרי גיגי

]] 'אל

In [ ]:

## בדיקה וערכה

כאשר אמנו את המוכנה אז TensorFlow נתן לנו תוצאות זמן אמת על אימון המוכנה. ולפי מה שניתן לראות בתמונה המוכנה אצלחה לזהות 95% מין הקבצים בצורה טובה:

```
Epoch 27/32
6/6 - 0s - loss: 0.0040 - accuracy: 1.0000 - val_loss: 4.6964e-04 - val_accuracy: 1.0000
Epoch 28/32
6/6 - 0s - loss: 0.0040 - accuracy: 1.0000 - val_loss: 4.4573e-04 - val_accuracy: 1.0000
Epoch 29/32
6/6 - 0s - loss: 0.0039 - accuracy: 1.0000 - val_loss: 4.2347e-04 - val_accuracy: 1.0000
Epoch 30/32
6/6 - 0s - loss: 0.0039 - accuracy: 1.0000 - val_loss: 4.0357e-04 - val_accuracy: 1.0000
Epoch 31/32
6/6 - 0s - loss: 0.0038 - accuracy: 1.0000 - val_loss: 3.8473e-04 - val_accuracy: 1.0000
Epoch 32/32
6/6 - 0s - loss: 0.0038 - accuracy: 1.0000 - val_loss: 3.6698e-04 - val_accuracy: 1.0000
```

## ניתוח יעילות

מימוש של המודל התאורטי שהוצג לעיל כתוכנת מחשב הוא למעשה אלגוריתם בינה מלאכותית ללמידה חישובית ממשפחת הלמידה המונחית.

בניגוד לפיתוח תוכנה קלאסית, אשר מתמקד במציאת פתרון אלגוריתמי לבעיות, פיתוח בעזרת רשת עצבית מתמקד באימון הרשת, הזנתה במידע מתאים וניתוח הפתרון. משימות אלו הן אמנם מורכבות וקשות לפחות כמו הגישה המסורתית, אך לעיתים קרובות ניתן למצוא בעזרת טכניקה זו פתרונות או קירובים לבעיות קשות.

הדרך הפשוטה היא לאחסן את כל הפרמטרים במערכים: את הקלט והפלט מהרשת ניתן לאחסן במערך אחד ואת המשקולות בשני. לדוגמה, נוכל לאחסן את כל המשקולות במערך תלת-ממדי עם שלושת הפרמטרים הבאים:

```
Weights[layer_number, neuron_number, connection_number] (מספר שכבה, מספר נוירון, מספר חיבור).
```

הפלט (כמו גם הקלט) מכל נוירון במערך דו-ממדי עם הפרמטרים הבאים הוא:

```
Output[ layer_number, neuron_number] (מספר שכבה ומספר נוירון).
```

נשים לב כי השדות  $O(1,1)$  ו- $O(1,2)$  הם קלט הרשת, והשדה  $O(3,1)$  הוא הפלט.

נוכל לגרום למחשב לחשב את הפלט מרשת כזו על ידי אלגוריתם פשוט:

- אתחל את כל המשקולות ושדות הפלט - השדות  $O$  ו- $W$  להיות אפס.

- קבע את הקלט - שדות  $O(1,1)$  ו-  $O(1,2)$  לערכים הרצויים כאשר שכבה 1 היא הקלט.
- קבע את המשקולות הרצויות לרשת.
- חשב את הפלט מהרשת (בדומה לרשימת "חישוב פלט הרשת").

בפרייקט שלנו יעלות של יצרית bag of word : יצרית מליון למסמך בודד- במסך בודד יש  $n$  מילים. וידוע לנו שמבנה הנתונים שלנו הוא מילון. אשר פעולה של הופסה וחיפוש נעשים ב  $O(1)$  לכן ליצור מילון של כל המילים בקובץ זה  $O(N)$

מיזוג כל מילונים למילון אחד. יודע לנו יש במילון אחד  $n$  מלים וידוע שיש לנו  $k$  מלונים לכל קבוץ לכן פעלה של מיזוג לוקחת  $O(n)$  ולמזג את כל המילונים יחד לוקח  $O(n*k)$

חופש שם במערך השמות : במערך השמות הוא מערך ממיון אלפי האלף בת. במכרך השמות יש לנו  $n$  שמות לכן חיפוש משתמש בחיפוש בינארי לכן היעלות של היא  $\log(n)$ . אנו מקבלים  $k$  מילים בשביל למצאית שם. אנו עוברי מילה אחת אחת ובודקים עם היא שם. לכן היעלות האלגוריתמית היא  $k \log(n)$

## אבטחת מידע

בבנית התוכנה אנחנו התייחסו עליה בתור מוצר אמיתי. כדי לשמור על זכויות היוצרים ולעצור שימוש פירטי של התוכנה אנחנו משתמשים במערכת Digital rights management וספציפית ב Always-on DRM. בשביל שימוש בתוכנה המשתמש צריך להיות תמיד מחובר לבסיס נתונים Fire base אשר מוגן באבטחת מידע של Google וניתן לגישה על ידי מתך איחודי (רשיון), לכן כל עוד המשתמש מחובר לבסיס נתונים אנחנו יודעים בבדעות שיש לו ראשיון ואישור מאיתנו להשתמש בתוכנה (המשתמש "קנה" אותה באופן חוקי). אם התוכנה לא מחוברת לבסיס נתונים מתרחש Soft lock ולא ניתן להשתמש בתוכנה, משמה משתמש שאינו לקוח שלנו לא יציח להשתמש בתוכנה בלי להפוך ללקוח.

## מסקנות על הפרויקט

- המירכת הייתה פרויקט קונספט ליכולות של מכונה לומדת. לפי תצאות הפרויקט ניתן להסיק :
1. בניית מכונה דורשת הבנה עמוקה של מתמטיקה.
  2. יצרית מכונה דורשת בדיקה רבה של פרמטרים שונים. ודורשתניסוי וטעייה.
  3. עבודה על פרויקט משותף לימדה אותנו לחלק עבודה בין שני האנשים ולעבוד בצוות.
  4. תחום הלמידה המוכנה הוא תחום של העתיד ויוכל לספק פתרונות רבים לבעיות עתידיות

## פיתוחים עתידיים

היות הצלחה הגדולה שלנו בזיהוי הנתונים. וגם השגת כל המטרות שלנו. ניתן עכשיו מהנתונים שהפקנו לעביר אותם לאנליסטיקה ולהפוך עוד שלבים. לשבילים אוטונומים ללא שימוש באדם. לדוגמא ניתן לראות הכתבה הזאת : <https://www.geektime.co.il/trullion-nabs-3-5-m-seed/>. הישתמש באותו קונצפט בשביל לאמן מכונה.

## בבליוגרפיה

<https://stackoverflow.com/questions/51572429/crop-image-in-tensorflow-object-detection-api-and-display-it>

[https://www.youtube.com/watch?v=a1br6gW-8Ss&ab\\_channel=JayBhatt](https://www.youtube.com/watch?v=a1br6gW-8Ss&ab_channel=JayBhatt)

<https://docs.oracle.com/javase/8/javafx/api/toc.htm>

[https://www.youtube.com/watch?v=j-3vuBynnOE&ab\\_channel=sentdex](https://www.youtube.com/watch?v=j-3vuBynnOE&ab_channel=sentdex) <https://stackabuse.com/python-for-nlp-creating-bag-of-words-model-from-scratch/>

<https://www.youtube.com/watch?v=vT1JzLTH4G4&list=PLC1qU-LWwrF64f4QKQT-Vg5Wr4qEE1Zxk>

<https://stackoverflow.com/questions/51572429/crop-image-in-tensorflow-object-detection-api-and-display-it>

<https://www.youtube.com/watch?v=a1br6gW-8Ss>

<https://www.youtube.com/watch?v=j-3vuBynnOE>

<https://stackabuse.com/python-for-nlp-creating-bag-of-words-model-from-scratch/>

<https://www.youtube.com/watch?v=vT1JzLTH4G4>