**Assignment 4**

* שיר כהן, 315805168
* עמית שקרצ'י, 313278889

**להגשה – קישור ל-github + הוראות איך להריץ**

**נקודות ושאלות לליאור:**

מודלים להשוואה:

* בייס-ליין- רשת בודדת (סוג הרשת).
* המודל הנבדק – masksembles
* המודל המשופר- masksembles + גיזום

היפר-פרמטרים:

* N- מספר המסכות (= מספר הרשתות הסופיות)
* Learning rate
* מספר שכבות הקונובלוציה

מספר הרשומות להרצה- כרגע אנחנו מריצות על 320 רשומות (train, test, validation) מכל דאטאסט (הדאטאסטים הטבלאיים שקיבלנו בעבודה קטנים יחסית). אחוזי הדיוק לא מספקים.

1. לכתוב מודלים -
2. לעדכן אופטימיזציה + לעדכן גודל רשת עבור N - עמית
3. להגדיל כמות רשומות
4. לעדכן את השטח pr
5. לבדוק גרפים של הבייאסיאני – לעשות ל-fold אחד בלבד
6. להוסיף שני dataset
7. לסדר את הקוד

# PAPER

**Masksembles for Uncertainty Estimation**

[**מאמר**](https://arxiv.org/pdf/2012.08334.pdf)

**IntroDUCTION**

הערכת האמינות ויכולת החיזוי של רשתות נוירונים עמוקות, היא נושא המחקר של המאמר. הכותבים מתייחסים ל-MC-Dropout ול-Deep Ensembles בתור פתרונות פופולריים לקבלת חיזוי אמין. שתי השיטות משתמשות ב-ensembles כדי להעריך את מידת אי הוודאות בחיזוי.

**MC-Dropout**- אימון של רשת אחת, בה מבצעים dropout בזמן האימון וגם בזמן הבדיקה של המודל. את החיזוי ניתן לעשות מספר פעמים, ומשום שאנו מבצעים dropout- נקבל חיזויים שונים בכל פעם. ביצועי השיטה ירודים יחסית במשימות של הערכת אי וודאות בשל המתאם הגובה בין הרשתות השונות ב- ensemble.

**Deep Ensemble**- בניית אנסמבל של מספר מודלים בלתי תלויים, וחישוב החיזוי על ידי ממוצע של כל התוצאות מהמודלים. ביצועי השיטה טובים יחסית במשימות של הערכת אי וודאות, אך זמני האימון וה-inference גדולים והשימוש בזיכרון רחב מאוד.

כותבי המאמר מציגים את **Masksembles**, גישה להערכת אי ודאות עם ביצועים זהים ל- Deep Ensemble, אך בעלויות חישוב נמוכות יותר. בשיטה זו, מבצעים dropout לפרמטרים של המודל בצורה מובנית (ולא רנדומלית כמו ב-MC-Dropout).

בגישה זו, משתמשים במספר מוגדר מראש של binary masks, לפיהם נבחרים הפרמטרים להם יבוצע drop. בזמן האימון- בוחרים mask רנדומלית, בדומה ל-dropout רגיל. במהלך ה- inference, מריצים את המודל מספר פעמים- כמספר ה-mask-ים כדי לייצר סט של פרדיקציות והערכת אי וודאות.

השיטה מקבל מספר פרמטרים:

* מספר ה-mask-ים הכולל.
* ממוצע של חפיפה בין mask-ים שונים
* מס' ה-1 וה-0 בכל mask.

הגדרת הפרמטרים יוצרת מודל שנע בין Deep Ensemble לבין MC-Dropout.

**METHODS**

[**קוד**](https://github.com/nikitadurasov/masksembles)

עבור dataset שמוגדר כאשר X מציין את הקלט ו-Y את הפלט. עבור בעיית חיזוי ממדלים את ההתפלגות כ-

כאשר זה המשקל של אלמנט k ב- ensemble ו-N זה מספר המודלים ב-ensmble.

עבור deep ensemble, המודלים הם בלתי תלויים ואינם חולקים משקולות משותפים ולכן הוא סט משקולות נפרד המאומן באופן עצמאי. ואילו עבור MC-Dropout יש משקל משותף כאשר מגדירים שהינו מסכה בינארית אקראית שנדגמת בכל איטארציה t.

*הבעיה בערך שבתהליך האימון יכולה להיות חפיפה עבור ערכים אלו ולכן יש קורלציה גבוהה במודל MC-Dropout. בנוסף, מאחר ובכל איטארציה מוגדרת מסכה אחרת חייבת להיות תגובה מתאימה בין כל ה-unit של הרשת ולכן מוביל למצב של אחידות בין החיזויים של מסכות שונות.*

*כדי להתמודד עם הבעיה הנ"ל הכותבים מציעים שיטה שבה מוגדרים מראש סט סופי של מסכות בינאריות כך שניתן לשלוט בחפיפה בניהן כדי להפחית את הבעיות שתוארו מעלה.*

***יצירת המסכות –***

N – *מספר המסכות*

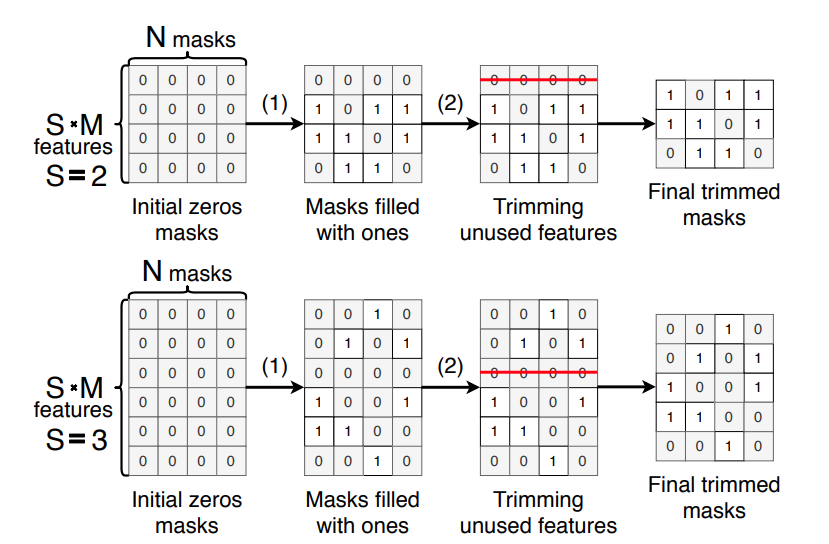
M – *מספר האחדות בכל מסכה*

S – *סקאלה ששולטת במספר החפיפות בין N ו-M.*

*אלגוריתם*

1. *יוצרים N ווקטורים בגודל MxS.*
2. *באופן אקראי מוגדר ה- M (מספר האחדות בווקטור) בכל ווקטור.*
3. *מסירים שורות שכולן מכילות אפסים.*

*עבור S=1 האלגוריתם מייצר N מסכות בגודל M שכולן יכילו אחדות ולכן יהיו חופפות לחלוטין. ואילו עבור ערכי S חיוביים גדולים יותר האלגוריתם ייצור N מסכות בעלות אפס ערכים רבים יותר ולכן מעט מאוד ערכים יהיו חופפים.*



**Experiments**

*כותבי המאמר ביצעו ניסויים עם ה- dataset של CIFRA10 ו- ImagNet.*

*הם השתמשו במודל Wide-ResNet-16-4. הם הוסיפו את שכבת ה-mask לפני כל שכבת קונבולוציה ו- fully connected. ה-mask הוגדר עם N=4 ו- S=5. הם השוו את המודל שלהם לרשת נוירונים בודדת, MC-Dropout ו- deep ensemble.*

**advantages of the algorithm**

הגישה המוצגת במאמר טובה יותר מבחינת דיוק, כיול ו- out of distribution משיטת MC-Dropout. בהשוואה ל- deep ensemble לשיטה יש ביצועים דומים אך היא בעלת זמן אימון קצר יותר ודורשת פחות זיכרון. באמצעות הגדרת הפרמטרים M,S ו-N, הגישה הגישה מאפשרת "תנועה" על הספקטרום בין הביצועים של Deep Ensemble, לבין זמני הריצה של MC-Dropout. למשל, כאשר S שואף לאינסוף, אין לנו חפיפה כלל במסכות והתנהגות הרשת היא כשל Deep Ensembles. כאשר S=1, יש לנו חפיפה מלאה והתנהגות הרשת היא כשל MC-Dropout. הדבר נותן למתכנן הרשת את היכולת לייצר הערכת אמינות טובה יותר מזו ש-MC-Dropout מספק, בבזבוז של פחות מקום וזמן חישוב מאשר Deep Ensembles.

**DISadvantages of the algorithm**

* בכל פעם שנרצה לשנות את M, N או S, נצטרך לבצע התאמה למספר השכבות ברשת, על מנת למנוע מצב בו בפועל לא מתקיים dropout בכלל.
* שכבות ה- Masksembles אינ מצליחות להתמודד עם יחידות מידע גדולות, למשל תמונות בגודל 256X256 פיקסלים.

# IMPROVMENT

Give a detailed explanation of your proposed improvement and why you think it might improve the algorithms performance.

# well-known algorithm

להשלים

# Dataset

**DeAtils**

|  |  |  |  |
| --- | --- | --- | --- |
| **Description** | **Number of classes** | **Number of Samples:** | **Name** |
| The MNIST database of handwritten digits. | 10 | 70,000 | **Mnist** |
| images of beans taken in the field using smartphone cameras. | 3 | 1,295 | **Beans** |
| Binary 20x16 digits of '0' through '9' and capital 'A' through 'Z'. | 36 | 1,404 | **binary\_alpha\_digits** |
| The CIFAR-10 dataset consists of 60000 32x32 colour images in 10 classes, with 6000 images per class. | 10 | 60,000 | **cifar10** |
| Images of healthy and unhealthy citrus fruits and leaves. | 4 | 594 | **citrus\_leaves** |
| The Stanford Dogs dataset contains images of 120 breeds of dogs from around the world. This dataset has been built using images and annotation from ImageNet for the task of fine-grained image categorization. | 120 | 20,580 | **stanford\_dogs** |
| Cassava consists of leaf images for the cassava plant depicting healthy and four disease conditions | 5 | 9,430 | **Cassava** |
| Images of hands playing rock, paper, scissor game. | 3 | 2,892 | **rock\_paper\_scissors** |
| A large set of images of horses and humans. | 2 | 1,283 | **horses\_or\_humans** |
| The Dmlab dataset contains frames observed by the agent acting in the DeepMind Lab environment, which are annotated by the distance between the agent and various objects present in the environment. The goal is to is to evaluate the ability of a visual model to reason about distances from the visual input in 3D environments. | 6 | 110,913 | **Dmlab** |
| This dataset contains images of - Handwritten Bangla numerals | 10 | 6,000 | **Cmaterdb** |
| Stanford Online Products Dataset | 12 | 120,053 | **stanford\_online\_products** |
| The STL-10 dataset is an image recognition dataset for developing unsupervised feature learning, deep learning, self-taught learning algorithms | 10 | 13,1000 | **stl10** |
| A large set of images of flowers | 5 | 3,670 | **tf\_flowers** |
| A large set of images of cats and dogs. | 2 | 23,262 | **cats\_vs\_dogs** |
| UC Merced is a 21 class land use remote sensing image dataset, with 100 images per class. | 21 | 2,100 | **uc\_merced** |
| Kuzushiji-MNIST is a drop-in replacement for the MNIST dataset (28x28 grayscale, 70,000 images), provided in the original MNIST format as well as a NumPy format. | 10 | 70,000 | **kmnist** |
| The Oxford Flowers 102 dataset is a consistent of 102 flower categories commonly occurring in the United Kingdom. | 102 | 8,189 | **oxford\_flowers102** |
| This dataset consists of 101 food categories | 101 | 101,000 | **food101** |
| The DeepWeeds dataset consists of 17,509 images capturing eight different weed species native to Australia in situ with neighbouring flora.The selected weed species are local to pastoral grasslands across the state of Queensland. | 9 | 17,509 | **deep\_weeds** |
| EuroSAT dataset is based on Sentinel-2 satellite images covering 13 spectral bands and consisting of 10 classes with 27000 labeled and geo-referenced samples. | 10 | 27,000 | **eurosat** |

**PRE-PROCESS**

ביצענו נרמול של ה-data לטווח ערכים של 0 עד 1. בנוסף, ביצענו resize של התמונות לגודל (32, 32, 1).

# Evaluat

**Algorithm**

**פסאודו קוד –**

1. עבור כל אלגוריתם
   1. עבור כל Dataset
   2. ביצוע 10 cross validation וקבלת train\_val, test
      1. ביצוע 3 cross validation על train\_val
         1. ביצוע Bayesian Optimization עם 50 קריאות.
      2. אימון האלגוריתם על ה- x\_train\_val
      3. הערכה על ה-test

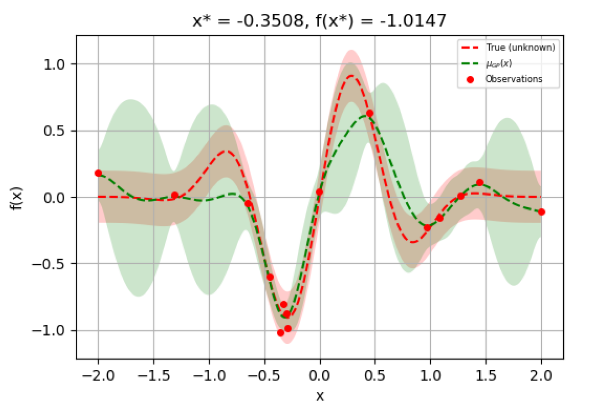
**Hyperparameter Optimization**

[**הסבר על השיטה Bayesian Optimization –**](https://github.com/mardani72/Hyper-Parameter_optimization/blob/master/Hyper_Param_Facies_tf_final.ipynb)

נרצה לבצע אופטימיזציה לפרמטרים של הרשת נוירונים. ניתן לחלק את האופטימיזציה לשני חלקים –

1. **פרמטרי אימון** – למשל המשקולות של רשת הנוירונים אשר נלמדים בתהליך האימון של האלגוריתם.
2. **Hyper-parameters** – פרמטרים שניתן לקבוע לפני האימון כמו מקדם הלמידה ומספר השכבות. הבחירה של הפרמטרים האלו היא מתישה, וקשה מאוד למצוא את הפרמטרים הטובים ביותר.

שיטת ה-grid search מבצע לולאה אשר עוברת על כל טווח הערכים האפשריים של הפרמטרים שהגדרנו. הבעיה בשיטה הנ"ל הוא זמן הריצה הארוך. למשל עבור 3 פרמטרים שלכל אחד מהם יש 10 אפשרויות השיטה תרוץ 10^3 פעמים. שיטה אחרת היא random search אשר לא עוברת על קומבינציות אלא מבצעת חיפוש אקראי אחרי האופטימיזציה האופטימלית. כאשר יש הרבה פרמטרים לכוונן שיטה זאת פחות יעילה.

השיטה שבה נשתמש היא Bayesian Optimization אשר מנסה למדל את ההשפעה של ערכי הפרמטרים על התוצאה שלנו [בהמשך להשלים איזה מדד בחרנו למקסם]. השיטה מניחה התפלגות גאוסיאנית. ציר ה-X מציין טווח של פרמטר שעליו עושים את תהליך ה-tuning, ציר ה-Y מציין את המדד. נרצה שהמודל שלנו יצליח להעריך את הפונקציה האדומה (הפונקציה האמיתית שבה הפרמטר שלנו מתנהג). הקו הירוק הירוק מייצג את הפונקציה שכרגע אנחנו מצליחים לייצר והשטח הירוק מראה את האי וודאות שלנו. ככל שהשטח גדול כך אנחנו פחות יודעים על האזור הזה. לאחר שנבדוק מספר נקודות ננסה להעריך מה הנקודה הבאה שמשתלם לבדוק.

החלטנו לאמוד את הפרמטרים הבאים – [להשלים]

הסיבה שבחרנו לאמוד את הפרמטרים היא [להשלים]

**Results**

הסבר על המדדים –

* **Accuracy – דיוק** – אחוז הדוגמאות שבהם האלגוריתם צדק.
* **Precision** – – מדד שמעיד כמה סיווגים "חיוביים" המודל הצליח לזהות בצורה נכונה. מדד זה עוזר לגלות מצבים שבהם ה- FP הוא גבוה (כלומר אמרנו כי ה-class הוא חיובי אבל בפועל הוא שלילי). יש מקרים שבהם נרצה לוודא כי אנחנו לא אומרים FP בתדירות גבוהה כמו להגיד על הודעה שהיא ספאם אבל בפועל הדוא"ל הוא רלוונטי. מאחר ואנחנו משתמשים ב-dataset של multiclass השתמשנו ב-flag "macro" אשר מחשב את ה-precision לכל label ומבצע ממוצע בין כל התוצאות.
* **Recall - TPR** - – מדד שמחשב כמה מהסיווגים ה"חיוביים" המודל הצליח לזהות. מדד זה עוזר לגלות מצבים שבהם ה- FN הוא גבוה (כלומר אמרנו שה-class הוא שלילי אבל בפועל הוא חיובי). יש מקרים שבהם נרצה לוודא כי אנחנו לא אומרים FN בתדירות גבוהה כמו להגיד לפציינט כי אינו חולה במחלה למרות שבפועל יש לו את המחלה. מאחר ואנחנו משתמשים ב-dataset של multiclass השתמשנו ב-flag "macro" אשר מחשב את ה-recall לכל label ומבצע ממוצע בין כל התוצאות.
* **FPR** - - שיעורי התחזיות החיוביות ב-class החיובי. כלומר, אחוז הדוגמאות שסיווגנו אותן כחיוביות אבל בפועל הן לא חיוביות. . מאחר ואנחנו משתמשים ב-dataset של multiclass השתמשנו בממוצע על פני כל ה-label.
* **AUC** – המדד מייצג את השטח מתחת לגרף של עקומת ROC. כלומר, AUC מחשב את השטח הדו ממדי מתחת לעוקמה שציר ה-X שלה הינו FPR וציר ה-Y הוא TPR. ככל שהמדד קרוב ל-1 כך התוצאה טובה יותר. מדד ה-AUC מייצג מדד מצטבר על פני כל ה-threshold האפשריים. המדד מייצג את ההסתברות שדוגמה חיובית אקראית תוגדר כדוגמה שלילית אקראית. החסרון של AUC שאינו מתחשב ב-threshold מסוים ולכן לא נוכל לשלוט לפי מדד זה ב- FP לעומת FN.
* **PR Curve** – Precision recall curve –גרף כאשר ציר ה-X הינו ה-recall וציר ה-Y הינו ה-precision. ביצענו את הגרף עבור כל label ב- dataset.
* **Training time** – זמן אימון של המודל. זמן האימון חושב עבור ה- X\_train\_val.
* **Inference time** – זמן חיזוי עבור 1000 דוגמאות.

טבלת תוצאות

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Inference  Time | Training  Time | PR Curve | AUC | Precision | FPR | TPR | Accuracy | Hyper Parameters  Values | Cross  Validation  [1-10] | Algorithm  Name | Dataset  Name |

**Statistical significance testing**

[**מבחן Friedman –**](https://drive.google.com/file/d/1-tx3kVk7J7LplN8lRTGRZ953vybL4oH-/view)

המבחן מאפשר להשוות אלגוריתמים על פני 20 ה-dataset. נחליט להשוות לפי מדד AUC שהתקבל מהממוצע של 10 cross validation שביצענו בשלבים הקודמים.

המבחן מגדיר דירוג של האלגוריתמים לפי המדד שקבענו. למשל עבור ה-dataset הראשון הוא ידרג את שלושת האלגוריתמים מהתוצאה הטובה ביותר עד התוצאה הגרועה ביותר כך שהאלגוריתם עם ה-AUC הגבוה יקבל דירוג 1.

הדירוג הממוצע יוגדר כ- כך ש-R מסמן דירוג ממוצע על פני כל ה-dataset ו-j מסמן את האלגוריתם.

L – מספר האלגוריתמים (3).

N – מספר ה-dataset שאנחנו בודקים (20).

*השערה האפס – לכל האלגוריתמים ביצועים זהים.*

*השערה האלטרנטיבית – קיימים אלגוריתמים עם ביצועים טובים יותר.*

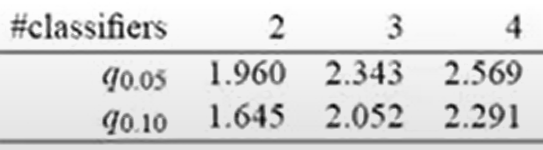
*רמת המובהקות שהגדרנו הינה 0.05.*

*לאחר הרצה של מבחן פרידמן על האלגוריתמים שלנו קיבלנו כי ה-pvalue שווה ל-– [להשלים] מאחר וערך זה קטן מ-0.05 אנחנו8 נעבור לבצע מבחן post hoc.*

*אם אנחנו דוחים את השערת האפס נעשה את מבחן* [*Post hoc*](https://www.statology.org/nemenyi-test-python/) *שמטרתו להשוות בין מסווגים ולהחליט האם אחד יותר טוב מהשני.*

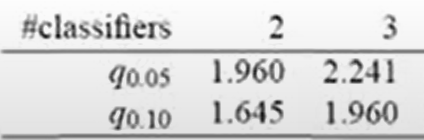
***אופציה ראשונה*** *– להשוות בין כל האלגוריתמים*

*נבצע הפרש בין הדירוגים של כל אלגוריתם ונבדוק אם ההפרש גדול מערך הסף –*



***אופציה שנייה –*** *להשוות את האלגוריתם המשופר שיצרנו לשני האלגוריתמים האחרים.*

*נבצע הפרש בין הדירוגים של כל אלגוריתם ונבדוק אם ההפרש גדול מערך הסף –*



קי*בלו מערך עם ה-value עבור כל אחד מהאלגוריתמים. להשלים את המערך*

*עבור ערכים קטנים מ- 0.05 נוכל להגיד ששתי הקבוצות שונות באופן מובהק סטטיסטי.*

# **conclusions**

Present the three algorithms you are evaluating (stages 1-3). Include the pseudo code, results and conclusions.