למידה חישובית יישומית

דוח פרויקט מסכם

קישור למאמר עליו התבססנו

https://www.researchgate.net/publication/280581078_Pseudo-Label_The_Simple_and_Efficient_Semi-Supervised_Learning_Method_for_Deep_Neural_Networks

קישור לתיקיית ה-Github של הפרויקט

https://github.com/ShamaySapir/Applied_Computational_Learning_Project

מגישות

ספיר שמאי 305649444

עינבר צור 312493927

1. הקדמה

מטרת התרגיל - הערכת ביצועים של שיטות Deep Learning ensamble שהתפרסמו בספרות המקצועית ולא סוקרו במהלך הקורס. התבקשנו לבחור מאמר, לתאר את האלגוריתם המוצע בו, את החסרונות והיתרונות שלו ולאחר מכן להעריך אותו באמצעות מדדים שונים אל מול אלגוריתם מוכר ואלגוריתם משופר. את ההערכה אנו מבצעות בעזרת 20 $datasets$שונים, במסגרת $10 - fold\ cross\ validation$ וכן בעזרת תהליך של $hyperparameter$ שנעשה במסגרת $3 - fold\ cross\ validation$. לבסוף אנו בודקות את ההשערה שקיימים הבדלים מובהקים סטטיסטית בעזרת $Friedman\ test$.

2. אלגוריתם להערכה

Pseudo-Label

מתוך המאמר:

Pseudo-Label : The Simple and Efficient Semi-Supervised Learning Method for Deep Neural Networks. Dong-Hyun Lee (2013)

2.1. תיאור האלגוריתם

שיטת ה-pseudo-labeling משתמשת בכמות קטנה של labeled Data וכמות גבוהה של unlabeled Data על מנת לשפר ביצועי מודלים מסוג Semi-Supervised Learning for Deep Neural Networks. האימון של המודל בכל epoch מתבצע על שני סוגי ה-data בו זמנית באופן הבא:

1. batch מאימון המודל שרץ על ה-labeled data ומחשב את ערך פונקציית ה-loss
2. batch שרץ על ה-unlabeled data ומשתמש במודל על מנת לבחור את הסיווג עם ההסתברות הגבוהה ביותר לכל רשומה
3. חיבור ה-labeled data ל-unlabeled data עם הסיווגים החזויים (Pseudo-Label), אימון המודל בשנית וחישוב ערך פונקציית ה-loss
4. שילוב ערך ה-loss של משלב 1 עם ערך ה-loss משלב 3 כך:

$$Loss\ per\ Batch\ =\ Labeled\ Loss\ +\ Weight\ *\ Unlabeled\ Loss$$

ערך ה-Weight משתנה כתלות בזמן. כלומר ככל שאנו מתקדמים ב-epochs נרצה לתת משקל גבוה יותר לערך ה-loss של ה-unlabeled data. במאמר מוצעת הפונקציה הבאה:

$$\alpha(t) = \begin{cases} 0 & t < T_1 \\ \frac{t-T_1}{T_2-T_1}\alpha_f & T_1 \le t < T_2 \\ \alpha_f & T_2 \le t \end{cases}$$

Equation [16] Lee (2013) [1]

כאשר:

- t הוא ה-epoch הנוכחי
- $\alpha(t)$- ערך ה-Weight
- T_1 ו-T_2 הם מספרי ה-epochs שקובעים את הגבולות בפונקציה לערך ה-Weight שיתקבל.
- α_f- ערך ה-Weight המקסימלי

2.2. מוטיבציה

שימוש ב-labeled data הוא יקר במונחי זמן וכסף. מצד שני, שימוש רק ב-unlabeled data נחשב עדיין כבעיה קשה. השילוב שכולל בתוכו חלק קטן של labeled data וחלק גדול יותר של unlabeled data הנקרא semi-supervised learning מתגבר על החסרונות הנ"ל. הרעיון של pseudo-labeling מציג שיטה פשוטה ויעילה שהציגה בניסויים ביצועי state-of-the-art.

2.3. יתרונות האלגוריתם

היתרון העיקרי של האלגוריתם הוא היכולת שלו לאפשר למודלי Deep Neural Networks להתאמן עם מעט labeled data ביחס ל-unlabeled data. עבודת התיוג עבור הבעיה של המודל נחסכת באמצעות האלגוריתם באופן משמעותי, משום שהוא מאפשר למודל לחזות סיווגים תוך כדי אימון המודל ועדכון המשקולות.

האלגוריתם הוא נאיבי ופשוט למימוש.

האלגוריתם יכול לשלב כמעט כל מודל neural network ושיטות אימון.

2.4. חסרונות האלגוריתם

האלגוריתם מבוסס על פונקציית משקל התלויה בזמן - epochs. הפונקציה מניחה שהביטחון של המודל עולה עם הזמן ולכן מגדילה את משקל פונקציית ה-loss של ה-unlabeled data באופן ליניארי. אך לעיתים תחזיות המודל עלולות להיות שגויות ולכן אם המודל מנבא מספר תחזיות שגויות ל-unlabeled data, ה-pseudo-label יכול לדרדר את ביצועי המודל.

בנוסף, כאשר כמות ה-labeled data הראשונית קטנה, הביצועים של תיוג ה-pseudo-label יורדים, הסיבה לכך היא שהאלגוריתם רגיש לסיווגים הראשונים ולכן נדרשת כמות מספקת של labeled data בשביל ביצועים טובים יותר.

2.5. פסאודו-קוד

```
function pseudo_label(model, labeled_data, unlabeled_data)
    epochs ← 1000
    α_f ← 3
    T_1 ← 100
    T_2 ← 600
    labeled_batch_size ← 32
    unlabeled_batch_size ← 256
    x_labeled_data, y_labeled_data ← split(labeled_data)
    y_unlabeled_data ← new array[unlabeled_data.size]
    final_losses ← new array[epochs]
    for epoch in epochs:
        model.fit(x_labeled_data, labeled_batch_size)
        loss_labeled ← loss(model.predict(y_labeled_data))
        y_unlabeled_data ← model.predict_proba(unlabeled_data)
        mix_data ←x_labeled_data + unlabeled_data
        model.fit(mix_data, unlabeled_batch_size)
        loss_unlabeled ← loss(model.predict(y_unlabeled_data))
        α_t ← 0
        if T_1 < epoch and epoch < T_2
            α_t ← (epoch − T_1)/(T_2 − T_1) *α_f
        else if epoch > T_2
            α_t ← α_f
        final_losses.append(loss_labeled + α_t * loss_unlabeled)
```

2.6. אופטימיזציה של Hyperparameter

המאמר מציע לבצע hyperparameter optimization על הפרמטרים הבאים:

- learning rate
- labeled data batch size
- unlabeled data batch size

לצורך ביצוע התהליך השתמשנו בספריית optuna. במחלקה Objective הגדרנו את הערכים האפשריים לכל פרמטר:

	The paper suggestion	Our suggestion in the code
learning rate	1.5	between 1.0 to 2.0
labeled data batch size	32	between 32 to 256
unlabeled data batch size	256	

המאמר מצא באמצעות תהליך hyperparameter את הערך 1.5 עבור ה-learning rate, אנו בדקנו את כל הערכים האפשריים בין 1 ל-2. המאמר מצא עבור ה-labeled data את גודל ה-batch להיות 32 ועבור ה-unlabeled data להיות 256. בעבודה אנו מתייחסות לסוג ה-data ברמת ה-epochs (מתואר בסעיף הבא 2.7) ולכן גודל ה-batch הוא יחיד ובודקות איזה ערך יהיה האופטימלי בין 32 ל-256.

2.7. תיאור מימוש האלגוריתם בעבודה

כעת נתאר את מימוש האלגוריתם, נציין כי בעבודה נעשו מספר הנחות.

כפי שראינו האלגוריתם יכול לשלב כמעט כל מודל neural network ושיטות אימון. במאמר הוצעה רשת נוירונים פשוטה הכוללת שכבה חבויה אחת כאשר:

- השכבת החבויה בעלת פונקציית אקטיבציה ReLU
- מספר ה-units בשכבה החבויה הוא 5000
- שכבת ה-output בעלת פונקציית אקטיבציה Sigmoid

אנחנו בנינו את הרשת הבאה:

- שכבת ה-input שכבת Dense:

$$Dense(12,\ activation\ =\ 'relu')$$

- שכבה חבויה ראשונה שכבת Dense:

$$Dense(8,\ activation\ =\ 'relu')$$

- מספר ה-units בשכבה זו הוא 8
- שכבה חבויה שנייה שכבת Dense:

$$Dense(10,\ activation\ =\ 'sigmoid')$$

- מספר ה-units בשכבה זו הוא 10
- שכבת ה-output שכבת Dense:

בעלת פונקציית אקטיבציה Sigmoid במידה וה-dataset מכיל משתנה מטרה בינארי

$$Dense(1,\ activation\ =\ 'sigmoid')$$

בעלת פונקציית אקטיבציה Softmax במידה וה-dataset מכיל משתנה מטרה multi-class

$$Dense(1,\ activation\ =\ 'softtmax')$$

דוגמא לרשת:

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 12)                108
_________________________________________________________________
dense_1 (Dense)              (None, 8)                 104
_________________________________________________________________
dense_2 (Dense)              (None, 10)                90
_________________________________________________________________
dense_3 (Dense)              (None, 3)                 33
=================================================================
```

שינוי נוסף שנעשה במימוש העבודה הוא היחס לסוג ה-data ברמת ה-epochs. כפי שראינו במאמר, בכל epoch ישנו מעבר על batch של labeled data ולאחר מכן מעבר batch על unlabeled data. במסגרת העבודה אנו מבצעות בכל epoch זוגי מעבר על labeled data לטובת אימון הרשת ובכל epoch אי זוגי מעבר על ה- unlabeled data לטובת הוספת pseudo-label ואימון מחדש יחד עם ה-labeled data וה-pseudo-label. אנו רצים עם 10 epochs (כלומר סך הכל 5 זוגות). באופן הזה אנו מנסות לשמר ככל הניתן את הרעיון של המאמר. בהתאם לשינוי הנ״ל ערך הפרמטר batch size נדרש להיות אחיד גם עבור labeled data וגם עבור unlabeled data מכיוון שהוא מוגדר ברמת ה-epoch, לכן הגדרנו לו טווח ערכים בין 32 ל-256 והפעלנו עליו את תהליך האופטימיזציה של hyperparameter.

חישוב פונקציית ה-loss מתבצע כפי שתואר במאמר כאשר:

$T_1 = 3$ ו-$T_2 = 8$ הם מספרי ה-epochs שקובעים את הגבולות

בפונקציה לערך ה-Weight שיתקבל.

$\alpha_f = 3$- ערך ה-Weight המקסימלי. חישוב

ערך ה-Weight בין ה-epoch ה-3 ל-8 הוא לפי החישוב הבא:

$$\frac{t-T_1}{T_2-T_1}\alpha_f \quad T_1 \leq t < T_2$$

ערך ה-loss הכולל מחושב בעבודה כממוצע של חמשת זוגות ה-epochs כך שלכל זוג epochs אנו מחשבים את ה-loss בדומה למאמר :

$$Loss\ per\ couple\ =\ Labeled\ Loss\ +\ Weight\ *\ Unlabeled\ Loss$$

2.8. פסאודו-קוד

```
function pseudo_label_work(model, labeled_data, unlabeled_data)
    epochs ← 10
    α_f ← 3
    T_1 ← 3
    T_2 ← 8
    batch_size ← hyperparameter(batch_size)
    x_labeled_data, y_labeled_data ← split(labeled_data)
    y_unlabeled_data ← new array[unlabeled_data.size]
    loss_labeled ← new array[epochs/2]
    loss_unlabeled ← new array[epochs/2]
    weights ← new array[epochs/2]
    final_losses ← new array[epochs/2]
    for epoch in epochs:
        if epoch % 2 == 0
            model.fit(x_labeled_data, labeled_batch_size)

loss_labeled.append( loss(model.predict(y_labeled_data)))
        else
            y_unlabeled_data ← model.predict_proba(unlabeled_data)
            mix_data ←x_labeled_data + unlabeled_data
            model.fit(mix_data, unlabeled_batch_size)

loss_unlabeled.append( loss(model.predict(y_unlabeled_data)))
            α_t ← 0
            if T_1 < epoch and epoch < T_2
```

$$\alpha_t \leftarrow (epoch - T_1)/(T_2 - T_1) * \alpha_f$$

$else\ if\ epoch\ >\ T_2$

$\alpha_t \leftarrow \alpha_f$

$weights.append(\alpha_t)$

$for\ i\ in\ range(5)$:

$final\_losses[i]\ \leftarrow (loss\_labeled[i]\ +\ weights[i]\ *\ loss\_unlabeled[i])$

3. אלגוריתם עם הצעת שיפור

בניסיון השיפור שלנו, ננסה להציע הצעה שתתגבר על שני חסרונות של האלגוריתם המקורי שתואר. החיסרון הראשון מתייחס לכמות הראשונית של ה-labeled data שגורם לביצועי התיוג להיות לא טובים מספיק. נציע שהרשת תאומן על ידי 4 epochs שכולם מורכבים מ-labeled data ובכך תאפשר ביצועים טובים יותר. בנוסף, החיסרון השני מתייחס למשקל ה-loss של ה-unlabeled data, שעולה ככל שעובר הזמן. ייתכן שהמודל ינבא מספר תחזיות שגויות ל-unlabeled data וה-pseudo-label ידרדר את ביצועי המודל, לצורך כך השינוי הנוסף יהיה לתת משקל שווה לכל חישוב loss. בתום כל epoch יחושב loss יחיד וערך ה-loss הכולל יהיה ממוצע של עשרת ה-epochs.

3.1 תיאור האלגוריתם

אלגוריתם pseudo-label עם השינויים הבאים:

- 4 epochs ראשונים יאמנו את המודל על בסיס labeled data בלבד ויחשבו את פונקציית ה-loss.
- ב-6 ה-epochs הנותרים המודל יסווג את ה-unlabeled data ויבצע אימון מחדש יחד עם ה-labeled data וה-pseudo-label ויחשב את פונקציית ה-loss.
- ערך ה-loss הכולל מחושב כממוצע של עשרת ה-epochs.
- הרשת זהה לזו שתוארה בסעיף 2.7

3.2. פסאודו-קוד

```
function improve_pseudo_label(model, labeled_data, unlabeled_data)
    epochs ← 10
    batch_size ← hyperparameter(batch_size)
    x_labeled_data, y_labeled_data ← split(labeled_data)
    y_unlabeled_data ← new array[unlabeled_data.size]
    final_losses ← new array[epochs]
    for epoch in epochs:
        if epoch < 4
            model.fit(x_labeled_data, labeled_batch_size)
            loss_labeled ← loss(model.predict(y_labeled_data))
            final_losses.append(loss_labeled)
        else:
            y_unlabeled_data ← model.predict_proba(unlabeled_data)
            mix_data ←x_labeled_data + unlabeled_data
            model.fit(mix_data, unlabeled_batch_size)
            loss_unlabeled ← loss(model.predict(y_unlabeled_data))
            final_losses.append( final_losses.append(loss_labeled))
    avg_loss ← avg(final_losses)
```

4. אלגוריתם ידוע-היטב

אימון מודל deep neural network באמצעות supervised learning.

4.1 תיאור האלגוריתם

- בתהליך ה-supervised learning נבדוק את ביצועי הרשת שתוארה בסעיף 2.7 על ה-labeled data בלבד, לאורך 10 epochs.
- ערך ה-loss הכולל מחושב כממוצע של עשרת ה-epochs.

4.2 פסאודו-קוד

$function\ well\_known(model,\ labeled\_data,\ unlabeled\_data)$

$epochs\ \leftarrow 10$

$batch\_size \leftarrow hyperparameter(batch\_size)$

$x\_labeled\_data,\ y\_labeled\_data\ \leftarrow split(labeled\_data)$

$y\_unlabeled\_data \leftarrow new\ array[unlabeled\_data.size]$

$final\_losses \leftarrow new\ array[epochs]$

$for\ epoch\ in\ epochs:$

$\quad model.fit(x\_labeled\_data,\ labeled\_batch\_size)$

$\quad loss\_labeled.append(loss(model.predict(y\_labeled\_data)))$

$avg\_loss\ \leftarrow avg(final\_losses)$

5. הערכת האלגוריתמים

נעריך את שלושת האלגוריתמים באמצעות המדדים הבאים:

- Accuracy – Under the assumption that the classification is the Class with the highest probability.
- TPR
- FPR
- Precision
- AUC – Area Under the ROC Curve
- Area under the Precision-Recall
- Training time
- Inference time for 1000 instances

5.1 טבלת מדדים

בטבלה כללנו את נתוני ה-datasets אשר הרצנו. לצערנו, הצלחנו להריץ את האלגוריתמים רק על חלק מה-20 datasets ולכן נציג בטבלה את הנתונים עבור ה-datasets שהרצנו.

6. בדיקת מובהקות סטטיסטית

השערת ה-0: ביצועי שלושת האלגוריתמים זהים

השערה אלטרנטיבית: לפחות אלגוריתם אחד הוא בעל ביצועים טובים יותר מהאחרים

מבחן Friedman - השוואה בין יותר משני אלגוריתמים על מספר רב של datasets.

נשתמש במדד ה-Accuracy, לצורך כך נחשב את הממוצע של המדד עבור כל dataset (ממוצע עבור 10 ריצות ה-cross validation)

את החישוב ביצענו עבור ה-datasets אשר הרצנו בשלוש האלגוריתמים, כדי לשמור על מהימנות התוצאות.

dataset	Pseudo-Label Semi-Supervised	Improved Pseudo-Label Semi-Supervised	Supervized
abalon	0.537693222	0.565795432	0.33492984
breast-cancer-wisc.csv	0.740311943	0.761380867	0.64402597
breast-cancer-wisc-prog.csv	0.814871795	0.829846154	0.785
breast-cancer-wisc-diag.csv	0.743409481	0.767805663	0.61980676

כעת נגדיר את הדירוג לכל אלגוריתם פר dataset ונחשב את הממוצע

לכל אלגוריתם:

dataset	Pseudo-Label Semi-Supervised	Improved Pseudo-Label Semi-Supervised	Supervized
abalon	2	1	3
breast-cancer-wisc.csv	2	1	3
breast-cancer-wisc-prog.csv	2	1	3
breast-cancer-wisc-diag.csv	2	1	3
Average Rank	2	1	3

נחשב את הקריטי כאשר:

$L = 3,\ N = 4$

$$\chi_F^2 = \frac{12N}{L(L+1)}\left[\sum_j R_j^2 - \frac{L(L+1)^2}{4}\right]$$

$$F_F = \frac{(N-1)\chi_F^2}{N(L-1)-\chi_F^2}$$

$$\chi^2_F = \frac{12*4}{3*4}\left[1 + 4 + 9 - \frac{3*16}{4}\right] = 8$$

$$F_F = \frac{3*8}{4*2-8}$$

קיבלנו שערך המבחן מתחלק ב-0.

אולם ניתן לראות שהאלגוריתם המשופר, הראה תוצאות טובות יותר עבור ה-datasets שנבדקו.

7. סיכום ומסקנות

בעבודה התנסנו בכתיבת אלגוריתם בתחום Deep Learning ensamble.

ביצענו מספר ניסויים גדול על האלגוריתם, על שיפור שלו וכן על אלגוריתם מוכר. לאחר מימוש האלגוריתם אשר הוצע במאמר ראינו כי השילוב של אימון המודל על מידע מתויג בשילוב עם מידע שאינו מתויג הביא לשיפור המודל שהוצא במאמר. לאחר המסקנה כי שילוב בין מידע מתויג ללא מתויג הביא לשיפור בתוצאות, הצענו אלגוריתם אשר מאמן תחילה "שלב חימום" על מידע מתויג בלבד לאחר מכן ביצענו pseudo label למידע שאינו מתויג והמשכנו את אימון המודל על המידע המשולב: מידע מתויג ומידע שתייגנו ב-pseudo label. ראינו כי שלב ה"חימום" של המודל הביא לשיפור בביצועיו במודל הסופי אשר שילב מידע מתייג ושאינו מתוייג מתוך הניסויים שהתאפשרו לנו לבצע מצאנו כי האלגוריתם המשופר שהצענו הציג את התוצאות הטובות ביותר.