פרק 1 – מבוא המציג את האתגרים העיקריים וכן מושגים בסיסיים המשותפים לכל הפרקים הבאים.

פרק 2 – ***Deep Learning Face Representation from Predicting 10,000 Classes***

פרק 3 – ***DeepFace: Closing the Gap to Human-Level Performance in Face Verification***

פרק 4 – ***Deep Learning Face Representation by Joint Identification-Verification***

פרק 5 – ***FaceNet: A Unified Embedding for Face Recognition and Clustering***

פרק 6 – **Deep Face Recognition**

נספחים

<u>מבוא</u>

פרק קצר זה מציג את האתגרים העיקריים וכן מושגים בסיסיים המשותפים לכל הפרקים הבאים, ההסברים בפרק זה מוסברים בצורה <u>כללית</u> בלבד ובפרקים הבאים ההסברים נעשים יותר מעמקים. עם זאת המושגים המוצגים בפרק זה מהווים אבני בסיס לפרקים הבאים.

בעיית הזיהוי הפנים נחלקת לרוב לשתי בעיות:

1) **אימות פנים**

בהינתן תמונת פנים מטרת המערכת היא לאמת אם האדם הוא אכן מי שהוא טוען שהוא. וזה למעשה בעיית 1:1 . לדוגמא מערכת של אימות פנים יכולה להיות מותקנת על דלת כניסה של בית וכאשר בן אדם יעמוד מול דלת הכניסה הדלת תפתח רק אם אותו אדם מורשה להיכנס לבית.

2) **זיהוי פנים**

כאשר נתון לנו מערך נתונים המכיל k אנשים, נרצה שבהינתן תמונת פנים המערכת תחזיר לנו את השם של האדם בתמונה בהנחה והוא קיים במערכת. כלומר בעיית $1:k$

שתי הבעיות קשורות אחת בשנייה ובעיית האימות מהווה אבן בסיס לבעיית הזיהוי.

עבודה זו מציגה חמישה מאמרים העוסקים בבעיית הזיהוי והאימות על-ידי למידה עמוקה. המאמרים נבחרו בקפידה רבה ובאופן כזה שהמאמרים הם מאמרים מוכרים בתחום ומצוטטים מספר רב של פעמים וגם פורסמו אחד אחרי השני בציר הזמן ככה שכמעט כל שיטה מהווה שיפור לשיטה הקודמת לה.

אחד האתגרים של זיהוי פנים הוא בעיית $one\ shot\ learning$, כלומר ברוב מערכות זיהוי הפנים אנחנו צריכים להיות מסוגלים לזהות או לאמת את האדם בתמונה **בהינתן תמונת אימון אחת בלבד**.

בשביל להיות יותר ברור אדגים את הבעייתיות בדוגמא: נניח שיש לנו שלוש תמונות במערך נתונים/אימונים (התמונות משמאל) ותמונת קלט(התמונה מימין). המערכת צריכה לזהות שתמונת הקלט אכן נמצאת במערך נתונים ולעשות זאת כאשר יש לנו רק תמונה אחת של אותו אדם שניתן ללמוד ממנה. ברור שסיווג בעזרת softmax לא ייתן תוצאות טובות בגלל המחסור בנתונים.




בשביל להתמודד עם הבעיה נרצה שהרשת נוירונים תלמד פונקציות דמיון:

$$d(img1, img2) = degree\ of\ difference\ between\ images$$

פונקציה זו תקבל שתי קלטים ותיתן כפלט את ההבדלים שבין התמונות באופן כזה שאם בקלטים מופיע אותו אדם ערך הפלט יהיה קטן ואם בקלטים יש אנשים שונים נקבל ערך גדול. ולאחר מכן ניתן לקבוע ערך סף τ ולהגדיר:

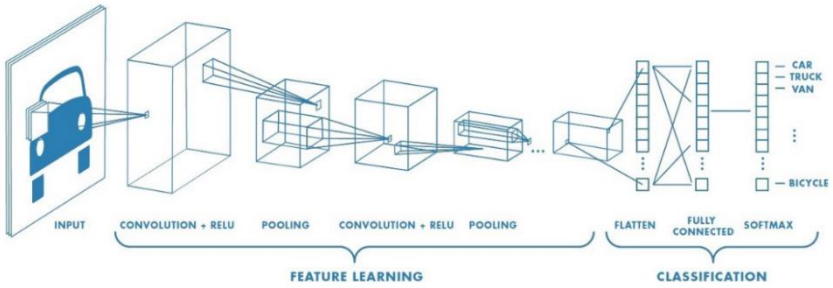
$$\begin{cases} if\ d(img1, img2) \leq \tau, & \text{same} \\ if\ d(img1, img2) > \tau, & different \end{cases}$$

גישה זן מתאימה לבעיית האימות, ובשביל בעיית הזיהוי ניתן להשתמש בפונקציה זו בשביל להשוות את תמונת הקלט לכל התמונות במערך נתונים, ואם קיבלנו הבדלי דמיון מספיק קטנים בין הקלט ובין תמונה כלשהי במערך נגיד שהאדם זוהה. במילים אחרות, הפונקציה מקבלת שתי קלטים ואומרת לנו עד כמה הקלטים דומים או שונים אחד מן השני.

בשביל ללמוד את פונקציית הדמיון נשמש ברשתות סיאמיות. רשתות הסיאמיות מתוארות במאמר *Siamese Neural Networks for One-shot Image Recognition*, ואסביר את מה שצריך לדעת להמשך הפרקים.

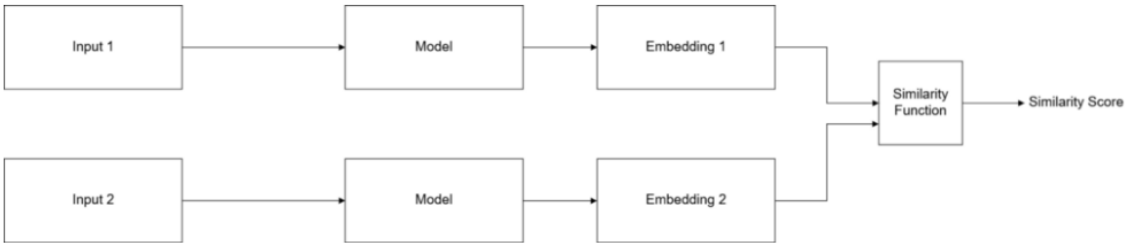
רשת קונבלציה פשוטה לבעיית סיווג מורכבת לרוב מרצף של שכבות קונבלציה ושכבות pooling ולאחר מכן ישנם שכבות מחוברות לחלוטין ושכבת פלט(כמובן שיש עוד סוגי שכבות ופעולות וציינתי את הבסיסיות שבהן בשביל לשמור על פשטות ההסבר). מטרת שכבת הקונבולציה היא לחלץ תכונות מהתמונה, כאשר השכבות הראשונות ברשת מחלצות לנו

תכונות בסיסיות(ברמה נמוכה) וככל שנעמיק בשכבות השונות ברשת נקבל תכונות יותר גלובליות (ברמה גבוהה). לאחר מכן בשכבת הפלט נקבל n הסתברויות(כאשר n מספר המחלקות שרוצים לסווג אליהם) המציינות את ההסתברות שהתמונה שייכת למחלקה ה-n-ית במערך אימונים.



Traditional CNN Architecture by Sumit Saha

רשתות סיאמיות מורכבות משתי רשתות קונבלציה **זהות** והיא מקבלת שתי קלטים בהתאמה. כל תת רשת בה מורכבת משכבות המבצעות פעולות שונות על תמונת הקלט, אך בשונה מהרשת שהוצגה קודם לכן בכל תת רשת במקום להתרכז בהסתברות שהתמונה שייכת למחלקה כלשהי, נתרכז בשכבה שלפני שכבת הפלט ונכנה את השכבה הזו ווקטור $Embedding$. כלומר כל תת רשת ברשת הסיאמית תקבל כקלט תמונה ובמקום להפיק הסתברות היא תפיק ווקטור שמהווה ייצוג/קידוד לתמונת הקלט.



נרצה לאמן את הרשת ככה שבהינתן שתי תמונות פנים $input_1$ ו-$input_2$ הרשת תפיק $Embedding_1$ ו- $Embedding_2$ בהתאמה, כך שלאחר מכן נהיה מסוגלים להגיד על בסיס אותם ייצוגים האם הקלטים שייכים לאותו אדם או לשתי אנשים שונים.

למשל לאחר הוצאת $Embedding_{1,2}$ נרצה להגדיר את פונקציות הדמיון להיות:

$$d(input_1, input_2) = \|Embedding_1 - Embedding_2\|_2^2$$

כלומר להיות המרחק בין ייצוגי הפנים של אותם קלטים ואז לקבוע ערך סף τ כך ש:

$$\begin{cases} if\ d(img1, img2) \leq \tau, & \text{same} \\ if\ d(img1, img2) > \tau, & different \end{cases}$$

ובאופן פורמלי נרצה:

$$For\ any\ input\ image\ x_i, parameters\ of\ NN\ define\ an\ \ Embedding_i.$$

$$Learn\ parameters\ so\ that:$$

$$If\ x_i, x_j\ are\ the\ same\ person, \left\|Embedding_i - Embedding_j\right\|^2\ is\ small$$

$$If\ x_i, x_j\ are\ different\ persons, \left\|Embedding_i - Embedding_j\right\|^2 is\ large$$

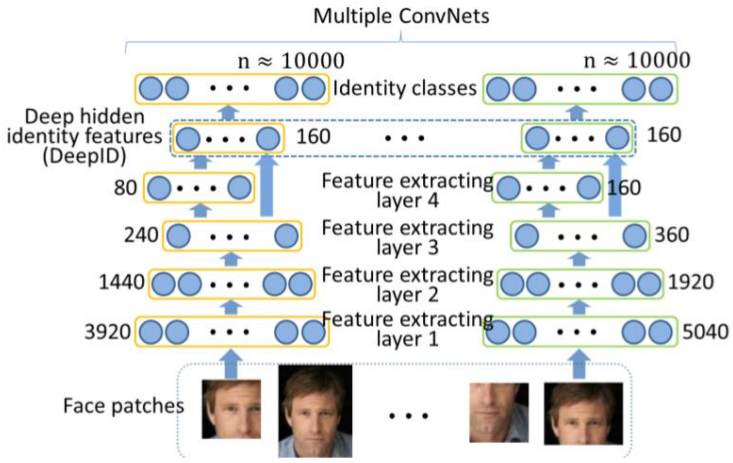
וכמו שנראה בהמשך רשת סיאמית יכולה להיות מורכבת מיותר משתי רשתות.

לסיכום, פרק זה הוא פרק מבוא והסברתי בו על מבנה הרשת הסיאמית והצגתי דוגמא על מה שעשוי להוות ייצוג טוב לתמונת פנים. בהמשך העבודה נראה פונקציות שונות בשביל לאמן רשתות ולאו דווקא סיאמיות שיתנו ייצוג נכון לתמונה פנים, כך שעל בסיסו נוכל לבצע השוואה.

Deep Learning Face Representation from Predicting 10,000 Classes

האלגוריתם המוצע במאמר זה הוא אלגוריתם להוצאת ווקטור תכונות מתמונת פנים, שנקרא DeepID. כלומר בהינתן תמונת פנים נרצה לקבל ווקטור תכונות ממד המייצג את הפנים, ולאחר שהוצאו ווקטורי התכונות במערך נתונים יהיה ניתן להשתמש בהם בשביל לבצע סיווג .

DeepID היא למעשה רשת קונבלציה שמאומנת עם שכבת softmax מעל $n \approx 10000$ מחלקות, כאשר כל מחלקה היא למעשה זהות/בן אדם במערך נתונים.



לכל תמונת פנים מוצאים 120 טלאים שונים(patches) והם מהווים קלט לרשת, ובהתאמה לכך הרשת עצמה בנויה מ-120 רשתות מחוברות.

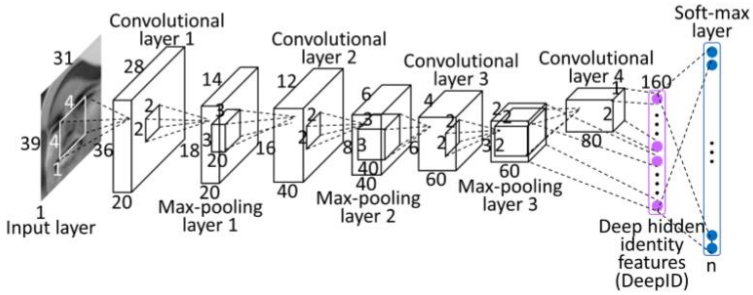
כל טלאי מתמונת הפנים עובר תהליך של הוצאת תכונות על-ידי ארבע שכבות קונבלציה ולאחר מכן עובר בשכבת מחוברת לחלוטין. השכבה המחוברת לחלוטין מכילה 160 נוירונים ובהתאמה מוציאה כפלט ווקטור תכונות באורך 160 המייצג את הטלאי. לאחר מכן ווקטור התכונות מחובר לשכבתsoftmax מימד $n \approx 10000$ אשר מוציא את ההסתברות להיות שייך לאחד מתוך n המחלקות.

לאחר שהרשת אומנה נשתמש בה בשביל לייצג תמונת ולשם כך נסיר את שכבת ה- softmax כך שנקבל שפלט הרשת יהיה ווקטור תכונות מימד 160 עבור כל טלאי, ומכיוון שכל תמונה מחולקת ל120 טלאים שונים נקבל שייצוג של תמונה אחת הוא ווקטור מימד 19200 (160×120).

האלגוריתם המוצע השיג דיוק של 97.45%.

הסבר זה מהווה הצגה כללית של השיטה המוצעת במאמר, ובמהלך הפרק אסביר את השיטה בצורה יותר מעמיקה ואת תוצאות השיטה.

מבנה הרשת מכיל ארבע שכבות קונבלציה ביחד עם שכבות $\max - \text{pooling}$ בשביל להוציא תכונות מהתמונה באופן הירכי, ולאחר מכן יש שכבה מחוברת לחלוטין ושכבת softmax לביצוע סיווג. שכבת הקלט מקבלת תמונות מימד $39 \times 31 \times k$ כאשר $k = 3$ עבור תמונות צבע ו-$k = 1$ עבור תמונה בגווני אפור. השכבה האחרונה קבועה להיות מימד 160 ומכונה שכבת DeepID וכמובן שממד שכבת הפלט משתנה בהתאם למספר המחלקות עליהם מבוצע הסיווג.



פעולת הקונבלציה מוגדרת להיות

$$y^{j(r)d} = \max(0, b^{j(r)} + \sum_i k^{ij(r)} * x^{i(r)})$$

כאשר:

- x^i – הקלט לשכבה ה-i. y^i
- k^{ij} – גרעין הקונבלציה בין השכבה ה- iלשכבה ה-j, ו-$*$ מסמנת את פעולת הקונבלציה.
- b^j - ערך ה- bias.

פעולת המקסימום בין פעולת הקונבלציה לבין 0 היא פונקציית אקטיבציה הנקראת גם Relu .

משקולות בשכבות העליונות של הרשת הם מחוברות לוקאלית בשביל ללמוד תכונות שונות באזורים שונים של התמונה

- r – מציין אזור מקומי בו המשקלים משתפים.

בשכבת הקונבלציה השלישית המשקולות משותפים באופן **לוקאלי** בכל אזור בגודל 2×2 בתמונה, ובעוד שבשכבה הרביעית אין כלל שיתוף בין המשקולות.

השכבה האחרונה **המוסתרת** של הרשת היא שכבה מחוברת לחלוטין והיא מחוברת לשכבת הקונבלציה השלישית וגם לרביעית , כך שהיא מקבלת תכונות שהם $multi - scale$. (כלומר היא מקבלת תכונות מסוימות מהשכבה השלישית ותכונות יותר גלובליות מהשכבה השלישית, ראה הסבר נוסף בנספח. ובנוסף מורידה את מספר הפרמטרים של הרשת) .

השכבה האחרונה היא בצעם שילוב לינארי בין התכונות של השכבות ולאחר מכן הפעלת פונקציית Relu. וניתנת לניסוח כ:

$$y_i = \max(0, \sum_i x_i^1 \cdot w_{i,j}^1 + \sum_i x_i^2 \cdot w_{i,j}^2 + b_j)$$

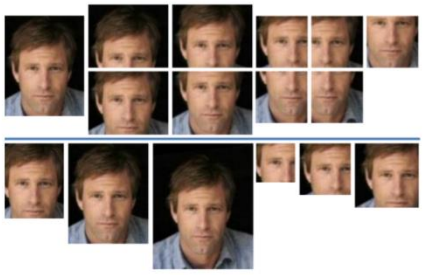
כאשר $x_i^1, w_{i,j}^1, x_i^2, w_{i,j}^2$ מציינים נוירונים ומשקולות בשכבה השלישית והרביעית בהתאמה.

שכבת הפלט היא $\mathrm{n-way\ softmax}$ אשר מוציאה כפלט הסתברות מעל $n \approx 10000$ מחלקות.

ונעשה שימוש ב-$Stochastic\ gradient\ descent$ עם $\mathrm{back-propagation}$ סטנדרטי לחישוב הגרדיאנטים.

תהליך הוצאת התכונות עבור תמונת פנים עובד באופן הבא: בהינתן תמונת קלט השלב הראשון הוא לאתר חמש נקודות ציון (facial landmarks), נקודה על כל עין, נקודה נוספת על האף ושתי נקודות על קצוות השפה. הוצאת נקודות הציון נעשית על-ידי אלגוריתם המתואר במאמר Deep Convolutional Network Cascade for Facial Point Detection. ולאחר מכן נעשה יישור של התמונה על-ידי טרנספורמציית דמיון על פי שני מרכזי העיניים ונקודות האמצע של פינות הפה.

לאחר מכן התמונה מחולקת ל-10 טלאים שונים, ולכל טלאי שקיבלנו נבצע scale ל-3 גדלים שונים, מה שייתן לנו 30 חלקים שונים של אותה תמונת פנים(באיור למטה ניתן לראות את החלוקה ל10 חלקים שונים ואת הגדלים). כל ה30 טלאים שקיבלנו הם RGB, אז נלקחו גם הגווני אפור של אותה תמונה ככה שקיבלנו שהכפלנו את מספר הטלאים וכעת אנחנו עומדים על 60 חלקים שונים של אותה תמונת פנים. לאחר מכן כל תמונה סובבה בצורה אופקית, וסך הכל קיבלנו 120 חלקים שונים של אותה תמונת פנים.



כתזכורת הרשת עצמה מורכבת מ-60 רשתות קונבלציה. **כל** רשת תקבל שתי קלטים של אותה תמונה כאשר הקלט הראשון הוא התמונה המקורית(או חלק ממנה) והקלט השני הוא אותה התמונה שסובבה בצורה אופקית(או חלק ממנה). וכל רשת תפיק שתי ווקטורי תכונות מימד 160 (בהתאם לסיבוב התמונה). ולכן ייצוג של תמונה נעשה בעזרת ווקטור באורך 19200 $(160 \times 60 \times 2)$ אשר ישמש אותנו לשלב האחרון של אימות פנים.

את **שלב אימות הפנים** ניתן לבצע בכמה שיטות, שתי השיטות המזכרות במאמר הם Joint Bayesian אשר מתוארת במאמר [Bayesian Face Revisited: A Joint Formulation] ובעזרת רשת נוירונים. ולאחר מכן בוצעה השוואה בין שיטות אלו.

בשביל לבצע אימות ברשת נוירונים ניקח כקלט שתי תמונות פנים ונוציא מהם את ווקטורי התכונות, והם ישמשו כלקט לרשת נוירונים לביצוע סיווג.

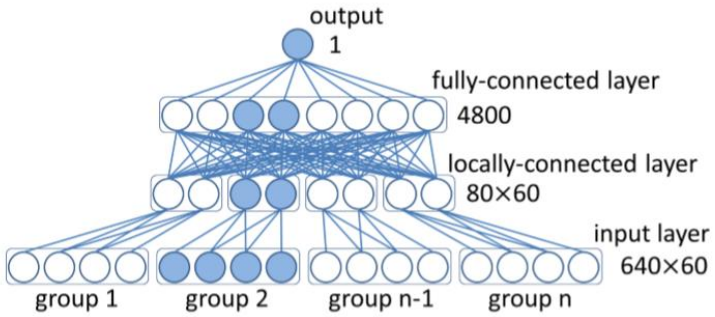
הרשת בנויה באופן הבא:

- **שכבת קלט** - המקבלת ווקטור תכונות המייצג תמונת פנים, כזכור הווקטור באורך 19200. הווקטור מחולק ל-60 קבוצות, קבוצה עבור על טלאי. כל קבוצה מכילה 640 תכונות שהוצאו מתוך זוג טלאים מסוים(בשלב הקודם ראינו שמספר התכונות עבור תמונה הוא יחידה הוא $160 \times 2 = 320$ ומכיוון שעושים שימוש בזוג תמונות $(160 \times 2 \times 2 = 640)$. סך-הכל אורך הקלט הוא 38400.

- **שכבה מחוברת לוקאלית** – נוירונים בשכבה זו מתחברים רק לקבוצה אחת של תכונות כדי ללמוד את הקשרים המקומיים ולהפחית את מימד התכונה בו זמנית. סך-הכל יש 60 קבוצות בשכבה זו, וכל קבוצה מכילה 80 נוירונים. סך-הכל 4800 נוירונים בשכבה זו.
- **שכבה מחוברת לחלוטין** – בכדי ללמוד קשרים גלובליים, ומכילה 4800 נוירונים.

- **שכבת פלט** – המכילה נוירון יחיד.

נעשה שימוש בפונקציית Relu בכל השכבות הנסתרות, ובשכבת הפלט השתמשו ב– sigmoid .

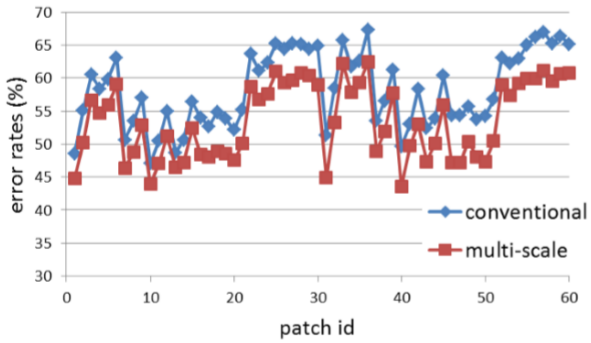


נעשה שימוש בשכבות Dropout (בשביל להימנע מהתאמת יתר) עבור כל שכבה נסתרת.

בשביל לאמן את הרשת השתמשו בערך הנתונים CelebFaces (המכיל 87628 תמונות פנים של 5436 אנשים שונים). ובשביל להעריך את התוצאות נעשה שימוש ב-LFW. **וחשוב להדגיש שבין המערכי נתונים אין כלל חפיפה.**

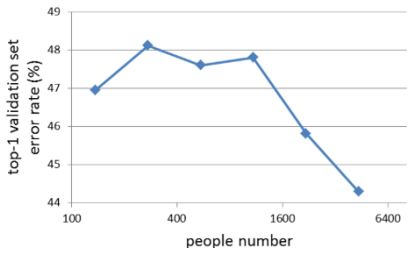
מתוך CelebFaces נבחרו באופן רנדומלי (4349) 80% מהאנשים בשביל לאמן את DeepID, כך שנקבל מודל שיחלץ לנו ייצוג של התמונות פנים. לאחר מכן עבור 20% הנותרים הוציאו ייצוג של התמונות בעזרת אותו DeepID מאומן וזה בכדי לאמן את המודל המבצע אימות. (רשת הנוירונים או Joint Bayesian) כלומר 80% ממערך האימונים שימש לאימון DeepID ושאר הנתונים שימשו לאימון המסווג. ולסט האימות של DeepID לקחו 10% מהתמונות (בצורה רנדומלית) מכל מחלקה. לאחר מכן בשביל ללמוד את ביצועי השיטה על מערכי נתונים גדולים יותר הם הרחיבו את מערך האימונים מCelebFaces ל-+ CelebFaces המכיל 202599 תמונות של 10177 אנשים שונים. חלוקת הנתונים נעשתה באופן זהה לCelebFaces מבחינת אחוזים (וגם במקרה זה אין חפיפה ל– LFW).

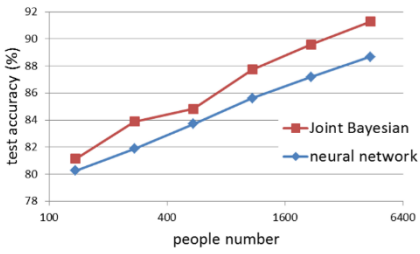
כזכור מבנה הרשת DeepID בנוי כך שהשכבה השלישית וגם השכבה הרביעית מחוברות לשכבה המייצגת את תכונות התמונה($multi-scale$) . באיור הבא ניתן לראות את אחוז השגיאות כאשר עושים שימוש $multi-scale$ ובין שלא(כאשר מאמנים את DeepID לבצע סיווג). כאשר שיעור שגיאות נמוך מצביע על כך שנלמדו תכונות טובות יותר.



שימוש $multi-scale$ מפחית את שיעור השגיאה ב– 4.72% בDeepID , מה שמשפיע באופן ישיר על הדיוק בביצוע אימות פנים בשיטת Joint Bayesian ומעלה את הדיוק מ– 95.35% (כאשר לא נעשה שימוש ב-$multi-scale$) ל-96.05%.

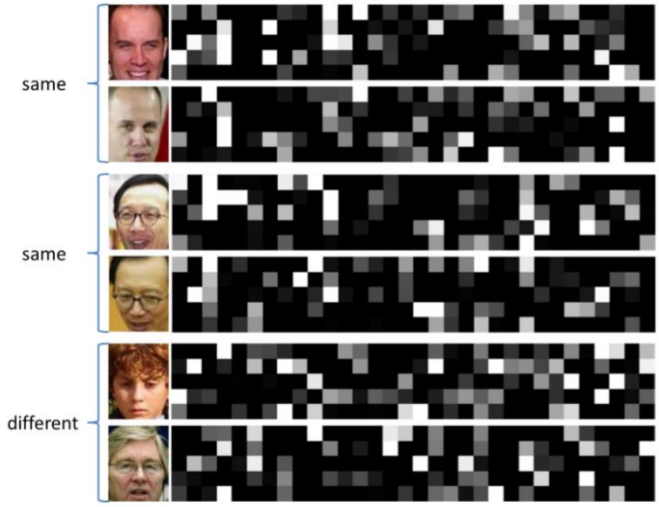
בנוסף אימנו את הרשת על מספר שונה של מחלקות בכל פעם כאשר בכל ניסוי העלו את מספר המחלקות בצורה אקספוננציאלית מ-136 מחלקות ועד 4349, שכמובן מספר הנוירונים בשכבת הסיווג שונתה בהתאם למספר המחלקות בעוד ששכבת התכונות באורך 160 נשארה ללא שינוי בכל הניסויים. בהתאם למספר המחלקות הדיוק עבור אימות פנים עלה ב-10.13% בJoint Bayesian וב-8.42% ברשת נוירונים. ובהתאמה לדיוק האימות שיעור השגיאה ירד.



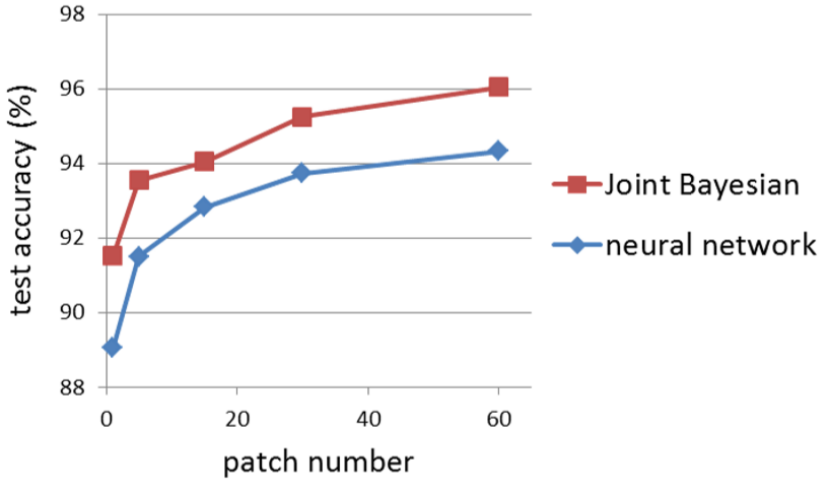


הגידול הלינארי של הדיוק ($test\ set$) ביחס לנתוני האימונים הגדלים באופן אקספוננציאלי מצביע על כך שהתכונות ישופרו עוד יותר אם יש יותר מקומות שאפשר להשיג נתונים.

נמצא שתכונות שמחולצות עבור זוג תמונות פנים שונות נוטות להיות דומות, בעוד שזוג תמונות שלא שייכות לאותו אדם נוטות להיות שונות. כדוגמא לכך ניתן לראות את התכונות שנלמדו באיור למטה. הרשת להוצאת תכונות אומנה קודם על 4349 מחלקות ולאחר מכן הוצאו תכונות של זוג תמונות מתוך LFW(שכזכור משמש כ$test\ set$) .



נבדק גם הקשר בין מספר הטלאים שהוצאו מתמונה לבין דיוק המודל לצורך אימות. אימנו את רשת האימות עבור k טלאים כאשר $k = 1, 5, 15, 30, 60$. דיוק המודל עלה ב-4.53% וב-5.27% כאשר נעשה שימוש ב-$k = 60$ במקום ב-$k = 1$ (Joint Bayesian ורשת נוירונים בהתאמה). וסך הכל הושג דיוק של 96.05% ו- 94.32% על Joint Bayesian ורשת נוירונים בהתאמה. (בהמשך נעשה שימוש בk=100)

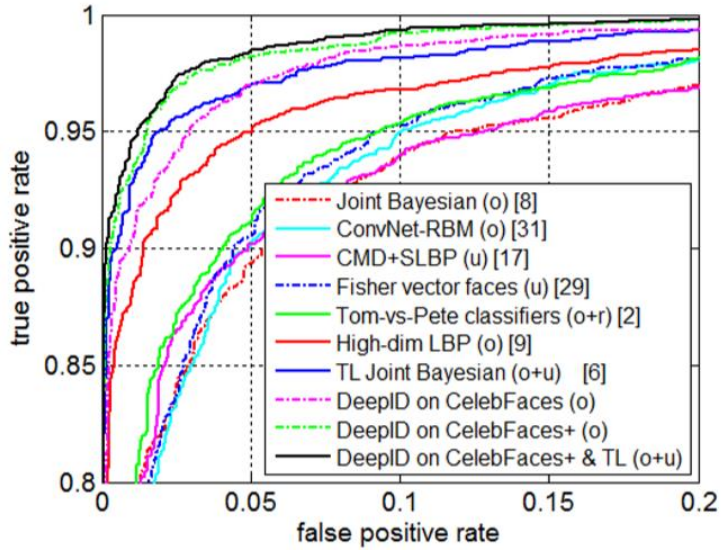


הניסוי האחרון נעשה בשביל להראות את ההשפעה של הוספת נתונים על דיוק השיטה. הם הרחיבו את מערך הנתונים ל+ CelebFaces המכיל 10177 מחלקות שונות ומתוכו נבחרו 8700 מחלקות לאימון DeepID ו1477 המחלקות הנותרות לאימון מודל האימות. מכיוון שלפני זה נעשה ניסוי שהראה ששימוש במספר רב יותר של טלאים משפר את דיוק האימות, בניסוי זה **הוצאו 100 טלאים** לכל תמונה. כתוצאה מכך נקבל ווקטור תכונות מימד 32000 שלאחר מכן בוצע PCA בשביל להוריד את מימד הווקטור ל150. על בסיס אותם ווקטורים אומן Joint Bayesian והפיק דיוק של 97.45% על LFW (שנאשר להיות test set), וזה בעוד שדיוק אדם עומד על 97.53%.

הטבלה הבאה משווה באופן מקיף את הדיוק שהושג ביחס לפרמטרים שונים, רק אזכיר שמספר הנקודות שהוצאו לכל תמונה משמש ליישור התמונה.

Method	Accuracy (%)	No. of points	No. of images	Feature dimension
Joint Bayesian [8]	92.42 (o)	5	99,773	2000 × 4
ConvNet-RBM [31]	92.52 (o)	3	87,628	N/A
CMD+SLBP [17]	92.58 (u)	3	N/A	2302
Fisher vector faces [29]	93.03 (u)	9	N/A	128 × 2
Tom-vs-Pete classifiers [2]	93.30 (o+r)	95	20,639	5000
High-dim LBP [9]	95.17 (o)	27	99,773	2000
TL Joint Bayesian [6]	96.33 (o+u)	27	99,773	2000
DeepFace [32]	97.25 (o+u)	6 + 67	4,400,000 + 3,000,000	4096 × 4
DeepID on CelebFaces	**96.05** (o)	5	87,628	150
DeepID on CelebFaces+	**97.20** (o)	5	202,599	150
DeepID on CelebFaces+ & TL	**97.45** (o+u)	5	202,599	150

ביצועי האלגוריתם שהוצג במאמר זה ביחס לאלגוריתמים אחרים השיג את התוצאות הכי גבוהות(עד אותו זמן) ובאיור למטה את ניתן את ההשוואה לאותם אלגוריתמים בעקומת ROC.



כסיכום: למדנו ייצוג של תמונות פנים בצורה סטנדרטית על-ידי softmax ולאחר מכן נעשה שימוש בייצוג בשביל לאמן מודל המצבע סיווג וזה בשביל להגיד אם שתי תמונות פנים שייכות לאותו אדם או לא. זה למעשה היה אחד המאמרים הראשונים שעשה שימוש ברשת קונבלציה עמוקה לבעיית זיהוי והאימות פנים, ולכן אף על פי שהושגו תוצאות ממש טובות הדרך עצמה הייתה סטנדרטית **ביחס לשיטות אחרות.** בהמשך נראה אלגוריתם בשם DeepID2 שמהווה שיפור לדרך שהוצגה במאמר וזה על-ידי תוספת לפונקציית softmax . בהמשך נראה גם שיטות שונות שלא עושות כלל ב- softmax .

DeepFace: Closing the Gap to Human-Level Performance in Face Verification

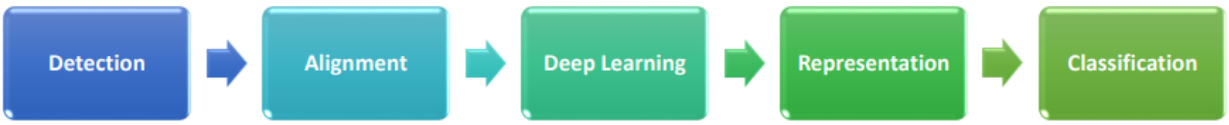
אני אתן סקירה **כללית** על מאמר זה ולאחר מכן ארד לפרטים.
מערכת DeepFace היא המערכת הראשונה שביצעה עבודה מוצלחת בנושא של אימות וזיהוי פנים וגם הראשונה שעשתה שימוש בלמידה עמוקה להשיג תוצאות אלו. המערכת פותחה על-ידי חוקרים מאוניברסיטת תל אביב ביחד עם חוקרים מפייסבוק, ולפי מיטב הבנתי פייסבוק עושה שימוש במודל זה לזיהוי פנים עד היום.

הדרך המוצגת במאמר זה בנויה מחמישה שלבי עבודה:

1. איתור פנים בתמונה.
2. יישור של תמונות.
3. אימון רשת קונבלציה.
4. ייצוג פנים כווקטור תכונות.
5. סיווג.

לא מעמקים בשלב האיתור פנים מכיוון שזה לא הדיון של המאמר, אבל כמובן שהוא חלק בלתי נפרד מתהליך העבודה. יישור התמונה(alignment) הוא חלק מהתהליך נרמול התמונות ומעלה את דיוק המערכת כולה, ובסוף הפרק ניתן לראות השוואות בין סוגי נרמול שונים או ללא נרמול כלל. מאמר זה מציג רשת קונבלציה ייחודית שפותחה עבור המשימה הזו, וזה בניגוד למאמרים אחרים שעושים שימוש ברשתות מוכרות. מטרת הרשת היא ללמוד ייצוג של תמונת פנים ולהפיק ווקטור תכונות המייצג אותה(שלב 4) , כך שבהינתן ווקטור תכונות של תמונת פנים השייכת לישות A ווקטור תכונות של ישות B יהיה ניתן להשוואות ביניהם.

לצורך אימות/זיהוי פנים הוצגו כמה שיטות מלבד השוואה בין ווקטורי התכונות, שיטה שעושה שימוש ב- SVM וב $\chi^2 -$ distance שיוסבר במהלך הפרק וברשתות סיאמיות.



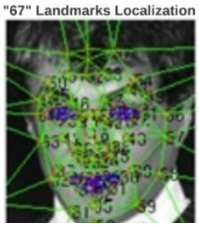
יישור של התמונות במערכי נתונים שונים עוזרים לשפר את הדיוק של הזיהוי בכך שהם מספקים מערך נתונים מנורמל. במאמר זה עושים שימוש במידול תלת-ממדי לפנים, שהמטרה שלו להביא תמונת פנים למצב חזיתי(frontalization). המידול מבוסס על מתארי fiducial points (ומעתה אקרא להם נקודות עוגן) בצורה איטרטיבית. בכל איטרציה מוצאים נקודות עוגן על-ידי שימוש ב- SVR שאומן לחזות נקודות אלו מתוך **מתארי** תמונות. מתארי התמונות שנעשה בהם שימוש הם LBP Histograms ,ומוזכר שניתן להשתמש במתארים אחרים. לאחר מכן מבצעים טרנספורמציה לתמונה בעזרת מטריצת דמיון (similarity matrix) לתמונה חדשה, ואז שוב מוצאים נקודות עוגן בתמונה החדשה.

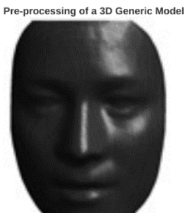
השלב הראשון: כולל בתוכו זיהוי של הפנים בתמונה(השיטה לכך לא תוארה במאמר, אני מניח שנעשה שימוש ב- viola jones algorithm). לאחר מכן מתבצע יישור בדו-מימד, וזה נעשה על-ידי מציאת שש נקודות עוגן כאשר יש נקודה אחת על האף, שתי נקודות על עיינים(נקודה עבור כל עין) ושלוש נקודת על הפה. נקודות אלו משמשות בשביל להעריך: קנה מידה(s_i) , סיבוב(R_i) והזזה (t_i) של התמונה לששה מיקומי עוגן על-ידי טרנספורמציה:

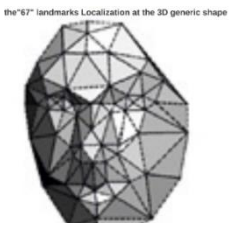
$$T_{2d} := (s_i, R_i, t_i)\ ;\ \text{where: } \mathrm{x}^{\mathrm{j}}_{\mathrm{anchor}} := s_i[R_i|t_i] * \mathrm{x}^{\mathrm{j}}_{\mathrm{source}}\ , \text{for points j } = 1..6$$

כאשר $\mathrm{x}_{\mathrm{source}}$ זו תמונה שמבצעים על-פיה את היישור. (צירפתי בנספח הסבר מוחשי על מה שקורה בפועל בשלב זה) בסוף תהליך זה נקבל תמונה מיושרת בדו מימד.

השלב השני הוא יישור בתלת-מימד, ולשם כך הדבר הראשון לעשות זה להוציא 67 נקודות אמון נוספות x_{2d} על התמונה שהופקה מהשלב הקודם ואת השילוש הדלוני שלהם. ולאחר מכן נעשה שימוש במודל תלת-ממדי גנרי ובהתמרה אפינית שתבצע עיוות(warp) לתמונה בדו-מימד למודל בתלת- ממד. המודל הגנרי שנעשה בו שימוש הוא ממוצע על תמונות הפנים מתוך USF Human-ID אשר התמונות פנים בו מיוצגות בצורה מיושרת ותלת-ממדית $v_i = (x_i, y_i, z_i)_{i=1}^{n}$, תויגו בצורה ידנית 67 נקודות עוגן על מודל זה x_{3d} .







מצלמה אפינית P(2x4) (כאשר הייצוג שלה נתון על-ידי וקטור $\vec{P}$ עם 8 דרגות חופש) מוגדרת לאחר מכן באמצעות פתרון generalized least squares למערכת הלינארית:

$$x_{2d} = x_{3d}\vec{P}$$

עם מטריצת שונות משותפת ידועה מראש Σ (covariance matrix) .

כלומר $\vec{P}$ ממזערת את ההפסד:

$$loss(\vec{P}) = r^T\Sigma^{-1}r \;;\; where: r = (x_{2d} - x_{3d}\vec{P}), and\ \Sigma\ is\ \text{covariance matrix}$$

נקודות שהתגלו על קווי המתאר של הפנים נוטות להיות יותר רועשות, מכיוון שמיקומן המשוער מושפע ברובו מהעומק ביחס לזווית המצלמה, משתמשים במטריצת שוניות Σ $(2*67)x(2*67)$

שניתנת על-ידי השונות המשותפת המשוערת של שגיאות נקודת העוגן. ניתן למזער את ההפסד בעזרת הפירוק של שולסקי של Σ, אשר מעבירה את הבעיה ל-ordinary least squares .

לאחר מכן ניתן להשיג ייצוג קנוני (frontalization) בכך שנבצע טרנספורמציה לכל משולש שייצרנו בתמונה בעזרת 67 נקודות עוגן.

המערכת לומדת ייצוג גנרי של תמונת פנים על-ידי רשת קונבלציה עמוקה.
הרשת מאומנת על כמה מחלקות (multi-class) של תמונות פנים, דהיינו עושה סיווג של תמונות פנים של אנשים שונים. הארכיטקטורה של הרשת היא כדלקמן:

שלב העיבוד המקדים:

(1) קלט – תקבל תמונת צבע בצורה תלת-ממדית ומיושרת באופן שבו הוצג קודם, שהגודל שלה הוא $(152x152)$

(2) **C1**– שכבת קונבלציה המכילה 32 פילטרים כאשר גודל כל פילטר הוא $(11x11x3)$ ולכן לאחר השכבה הזו נ קבל את הממדים $(142x142x32)$.

(3) **M2** – שכבת max – pooling שעושה שימוש בקרנל$(3x3)$ 3ו- stride שהוא 2 ונקבל את הממדים $(71x71x32)$

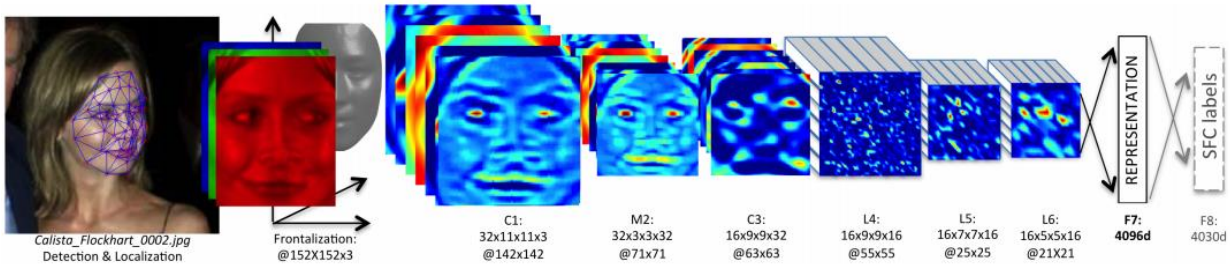
(4) **C3** – שכבת קונבלציה נוספת עם 16 פילטרים בגודל $(9x9x16)$ ונקבל את הממדים $(63x63x16)$.

מטרת השלוש שכבות שזה עתה תוארו היא להוציא low-level features, כמו קשתות וטקסטורות פשוטות. ושכבת $\max - \text{pooling}$ גורמת לרשת להיות יותר חסינה לשינויים לוקאלים ועם זאת כמה שכבות של $\max - \text{pooling}$ יגרמו לרשת לאבד מידע מדויק בתמונת פנים. ולכן ישנה רק שכבת $\max - \text{pooling}$ יחידה והיא בתחילת הרשת.
כותב המאמר קורא לשלוש שכבות אלו עיבוד מקדים, בגלל שיש להם מעט פרמטרים והם מרחיבות את הקלט לתוך קבוצה של תכונות מקומיות פשוטות.

השלב המרכזי :

(5) (6) (7) - הוא השלוש שכבות שלאחר מכן $(L4, L5, L6)$ שהם **שכבות מחוברות לוקאלית** שבשונה משכבת קונבלציה כל מיקום בfeature map לומד אוסף שונה של פילטרים והסיבה לכך היא שפילטר שרלוונטי למיקום מסוים בתמונת פנים כנראה לא רלוונטי למיקום אחר. בדוגמא לכך שכבה $L6$ ברשת מושפע מחלון בגודל $(74x\ 74x\ 3)$ בקלט, ואין כמעט שיתוף תכונות בין חלונות כאלה גדולים כאלה בפנים מיושרות. השימוש בשכבה זו אינו משפיע על החישובית של הוצאת התכונות, אבל אכן משפיע על מספר הפרמטרים. והסיבה שיש רק שלוש שכבות כאלה נעוצה בכך שיש מערך נתונים גדול. (צירפתי נספח המסביר את כמות הפרמטרים בשכבה זו בהשוואה לשכבת קונבלציה בסוף הפרק).

(8) (9) – שתי שכבות מחוברות בצורה מלאה (F8 F7) , כלומר יחידת פלט מחוברת לכל יחידות הקלט שבשכבה שלאחריה. שכבות אלו מסוגלות ללכוד את היחסים בין תכונות שנתפסו בחלקים מרוחקים של תמונות הפנים לדוגמה, מיקום וצורה של עיניים ומיקום וצורת הפה.



סה"כ הרשת כולל יותר מ120 מיליון פרמטרים כאשר 95% שייכים לשכבות $(L4, L5, L6)$

בדומה למה שהוסבר קודם לכן ייצוג בעזרת LBP כעת יהיה מיוצג על-ידי **הפלט של שכבה F7** כלומר פלט זה ישמש כייצוג של תמונת פנים כווקטור תכונות, ווקטור זה ישמש לסיווג.
הפלט של השכבה האחרונה מחובר ל-$\text{K} - \text{way softmax}$ (כאשר K הוא מספר המחלקות) המפיקה התפלגות מעל תוויות המחלקה.

אם נסמן ב - O_k את הפלט ה- k של תמונת קלט, אז ההסתברות שנקצה למחלקה ה-k תהיה הפלט של פונקציית softmax; כלומר:

$$\text{softmax function: } p_k = \frac{e^{O_k}}{\sum_h^k e^{O_h}}$$

מטרת האימון היא למקסם את ההסתברות של המחלקה הנכונה. ומשיגים זאת על-ידי צמצום של פונקציית $\text{cross} - \text{entropy}$ עבור כל דוגמת אימון. אם k הוא האינדקס של התווית המקורית עבור קלט מסוים, אז:

$$loss: L = -log(p_k)$$

פונקציית ההפסד ממוזערת על-ידי חישוב של הגרדינט של L ביחס לפרמטרים של הרשת ועל-ידי stochastic gradient descent (SGD). הגרדינטים מחושבים בעזרת backpropagation סטנדרטי. נעשה שימוש בפונקציית אקטיבציה:

$$\text{ReLU} = max(0, x)$$

לאחר כל שכבת קונבלציה, שכבה מחוברת לוקאלית, ושכבה מחוברת לחלוטין (מלבד השכבה האחרונה). בנוסף נעשה שימוש ב- dropout, שזו היא שיטה שמפחיתה את הסיכוי להתאמת יתר, על-ידי הפיכת חלק מהתכונות ל-0 בזמן האימון ובצורה רנדומלית (לא כתבו את הפרמטר שבו עשו שימוש בשיטה זו) . בגלל שעשו שימוש במערך נתונים גדול היה צריך ליישם שיטה זו רק עבור השכבה המחוברת לחלוטין הראשונה.

בהינתן תמונה I הייצוג $G(\mathrm{I})$ מחושב על-ידי העברה של התמונה ברשת(feed-forward) שזה עתה תוארה. כל העברה של תמונה ברשת עם L שכבות יכולה להיות מתוארת כהרכבה שך פונקציות $g_\emptyset^l$. ובמקרה שלנו הייצוג הוא:

$$G(I) = g_\emptyset^{F_7}(g_\emptyset^{L_6}(\ldots g_\emptyset^{C_1}(T(I, \theta_T))\ldots))$$

כאשר $\emptyset = \{C_{1,} \ldots, F_7\}$ הפרטמרים של הרשת ו- θ_T הייצוג שתואר בתחילת הפרק.

כשלב אחרון מבצעים נרמול ככה שהייצוג יהיה בין 0 לבין 1 במטרה להפחית את השינוי תאורה. כל ווקטור מחולק בערך המקסימלי שלו .
וזה נעשה על-ידי:

$$L_2 - \text{ normalization:} \qquad \mathrm{f(I)} := \frac{\bar{G}(I)}{\|\bar{G}(I)\|_2}\ ; \qquad \bar{G}(I)_i = \frac{G(I)_i}{max(G_i, \varepsilon)}$$

ובשביל למנוע חלוקה במספר קטן מדי ניתן לבחור $\varepsilon = 0.05$

בעיית אימות צריך להגדיר מדד כלשהו שימדוד אם שתי קלטים נתונים שייכים לאותה מחלקה, כלומר האם מדובר באותו אדם. וישנם שתי שיטות לכך, למידה מפוקחת ולמידה לא מפוקחת; כאשר השיטה המפוקחת משיגה תוצאות טובות יותר מהסיבה שאם מאמנים על מערך אימונים מסוים ניתן לעשות שינויים במודל בשביל לקבל תוצאות טובות יותר על אותו מערך אימונים. לדוגמא מערך האימונים LFW מכיל 75% מהתמונות שלו תמונות של גברים, ואימון על מערך אימונים זה יכול לגרום להכללה ככה שאם נבחן את תוצאות המודל על מערך אימונים אחר נקבל תוצאות לא טובות. מצד שני אימון על מערך אימונים קטן יכול לצמצם את ההכללה, ובמקרה זה השיטה הלא מפוקחת טובה יותר. במטרה למצוא האם שתי תמונות נתונות שייכות לאותה מחלקה מאמר זה בדק שלוש שיטות, שיטה אחת לא מפוקחת ושתיים בצורה מפוקחת. כאשר שלושת השיטות מסתמכות על הווקטור תכונות שנוצר במודל שהוסבר קודם לכן.

נסמן את וקטור התכונות שנוצר של התמונה הראשונה ב-$f1$ ואת הווקטור עבור התמונה השנייה ב-$f2$. **השיטה הלא מפוקחת** שנבדקה עשתה שימוש במכפלה הפנימית בין $f1$ לבין $f2$. התוצאה הסופית (האם אותו בן אדם או בן אדם שונה) תתקבל על-ידי ערך סף שהוגדר מראש.

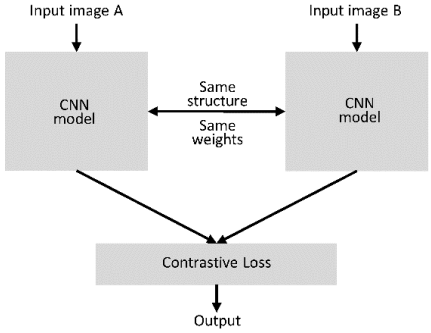
השיטה השנייה היא Weighted χ^2 distance כאשר הדמיון בין ווקטורי תכונות מוגדר להיות:

$$\text{weighted} - \chi^2 \text{ similarity: } \chi^2(f_1, f_2) = \sum_i \omega_i \frac{(f(x^{(1)}) - f(x^{(2)}))^2}{f(x^{(1)}) - f(x^{(2)})}$$

$$f(x^{(1)}), f(x^{(2)}) \in \text{DeepFace representations}$$

כאשר משקל ω נלמד בעזרת SVM לינארי שמוחל על $\frac{(f(x^{(1)}) - f(x^{(2)}))^2}{f(x^{(1)}) - f(x^{(2)})}$.

השיטה השלישית שנבדקה היא שימוש ברשתות סיאמיות, ואם להיות יותר מדויק אז על-ידי שימוש ברשתות סיאמיות **ובמסווג בינארי**. רשת סיאמית נראית כך:



כלומר היא רשת הבנויה משתי רשתות זהות לחלוטין, כלומר לשתי הרשתות **אותו מבנה** בדיוק **ואותם פרמטרים. ובהינתן שתי תמונות קלט הם יעברו ברשתות אלו ביחד יעברו את אותם החישובים בדיוק.**

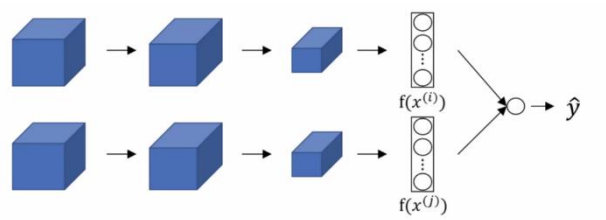
הרשת הסיאמית תחשב את המרחק הסיאמי בין שתי ייצוגי פנים כאשר המרחק הסיאמי מוגדר להיות:

$$d(f_1, f_2) = \sum_i \alpha_i \left| f(x^{(1)}) - f(x^{(2)}) \right|$$

$$f\left(x^{(1)}\right), f\left(x^{(2)}\right) \in \text{DeepFacerepresentations}$$

כאשר α_i הוא הערך שנלמד אותו. הפרמטרים של הרשת הסיאמית מאומנים בעזרת cross – entropy **ובעזרת** backpropagation. אם המרחק מתחת לסף מסוים אז נקבע שהפנים שייכים לאותו בן אדם, אחרת שונים.

המרחק מאומן על מערך האימונים אבל רק לאחר הקפאת כל השכבות מלבד שתי השכבות החדשות שנוספו.



כלומר דרך נוספת לאמן רשת לביצוע אימות זה לקחת את הרשת הסיאמית ככה ששתי הענפים של הרשת הסיאמית (כלומר הרשתות הזהות המרכיבות אותה) יחשבו את ווקטור התכונות של כל תמונה בנפרד, והווקטורים האלו יהוו קלט **ליחידת** רגרסיה לוגיסטית שנסמן ב-$\check{y}$, ונרצה ש:

$$\check{y} = \begin{cases} 0 & \textit{The same person} \\ 1 & \text{Different persons} \end{cases}$$

יחידת הרגרסיה לוגיסטית מוגדרת באופן הבא:

$$\check{y} = \sigma\left(\sum_{k=1}^{n} \omega_i \left| f(x^{(1)})_k - f(x^{(2)})_k \right| + b\right)$$

$$f(x^{(1)}), f(x^{(2)}) \in \text{DeepFace representations}$$

כאשר n זה אורך ווקטור התכונות שהופק, כלומר ערך k הוא בעצם המרכיב ה-kth בווקטור. וסה"כ

$$d(f_1, f_2) = \left|f(x^{(1)}) - f(x^{(2)})\right|$$
$$f(x^{(1)}), f(x^{(2)}) \in \text{DeepFace representations}$$

וכך נתייחס לאימות/ זיהוי פנים כבעיית סיווג לינארי.

המערכת כולה אומנה ונבחנה על שלוש מערכי נתונים עיקריים:

המערך אימונים הראשון שהמערכת אומנה עליו הוא **SFC** מערך אימונים זה נוצר על-ידי פייסבוק. והוא כולל 4.4 מיליון תמונות של 4030 אנשים שונים כאשר לכל בן אדם יש 800- 1200 תמונות. עבור המערך בדיקה (test set) הם לקחו 5% מהתמונות של כל מחלקה. המערכת אומנה כ-multi-class classification ונעשה שימוש בback-propagation סטנדרטי. בנוסף הפרמטרים שנעשו בהם שימוש הם:

$batch\ size = 128$, SGD with momentum $= 0.9$,epochs = 15

learning rate $= 0.01$. כאשר קצב הלמידה קטן בכל פעם עד שקצב הלמידה הגיע ל-0.0001 אתחול המשקולות בכל שכבה נעשה על-ידי zero-mean Gaussian distribution כאשר σ = 0.01 נעשה שימוש ב- GPU וזמן האימון הכולל ערך שלושה ימים. וכמו שהוסבר קודם לכן, שכבה F7 מחולצת כדי לשמש ייצוג לפנים.

המערך אימונים השני הוא LFW והוא מכיל 13323 תמונות מהאינטרנט של 5749 אנשים מפורסמים אשר מחולקים ל-6000 זוגות של תמונות ב-10 פיצולים שונים. הביצועים על מערך נתונים זה נמדדו בשלוש שיטות:

1. שיטה מוגבלת – כאשר המערך אימונים מורכב מזוגות של אותם אנשים או לא אותם אנשים.
2. שיטה לא מוגבלת - כאשר זוגות אימונים נוספים נגישים באימונים.
3. שיטה לא מפוקחת – המודל לא התאמן על מערך אימון LFW.

נמדדו ביצועים של כל מרכיב שתואר במאמר זה.

1. ללא formalization, כלומר בלי להביא את התמונה לצורה חזותית (בהתחשב בכך שחלקים מסוימים בתמונה עלולים לא להימצא כתוצאה של חיתוך התמונה לאחר זיהוי):
   - כאשר נעשה שימוש ביישור בדו-מימד הדיוק שהתקבל הוא 94.3%.
   - ללא יישור התמונה בכלל, כלומר רק על-ידי שימוש בתמונה חתוכה הדיוק עמד על 87.9%.
2. ללא אימון, ושימוש ב-formalization ובשילוב בין LBP לבין SVM, הושג דיוק של 91.4%.

כל מערך האימונים LFW עבר את אותו תהליך כמו שתואר במערך האימונים SFC , ונסמן את המודל ב DeepFace-single .

בשביל להעריך את הביצועים בשיטה לא מפוקחת נעשה שימוש במכפלה פנימית בין שתי ייצוגי פנים בצורה מנורמלת, ובמקרה זה הדיוק עמד על 95.92%.

ובשביל להעריך את השיטה המפוקחת נעשה שימוש ב SVM על $\chi^2 -$ distance והושג דיוק של 97.00%.

DeepFace-single	**0.9592** ±0.0029	unsupervised
DeepFace-single	**0.9700** ±0.0028	restricted

נבחנו גם שיטות נוספות שהוזכרו בפרק והתוצאות שלהם:

Network	Error (*SFC*)	Accuracy ± SE (*LFW*)
DeepFace-align2D	9.5%	0.9430 ±0.0043
DeepFace-gradient	8.9%	0.9582 ±0.0037
DeepFace-Siamese	NA	0.9617 ±0.0038

המערך אימונים השלישי הוא YTF, המכיל 3425 סרטונים של 1595 אנשים מפורסמים.
והמטרה של מערך אימונים זה הוא למדוד ביצועים של אימות פנים ברמת סרטונים. איכות התמונות במערך זה נמוכות יותר ממערכי האימונים אחרים כתוצאה מטשטוש בגלל תנועה או המרחק של הפנים בסרטון. התוצאות שהתקבלו הם:

Method	Accuracy (%)	AUC	EER
MBGS+SVM- [31]	78.9 ±1.9	86.9	21.2
APEM+FUSION [22]	79.1 ±1.5	86.6	21.4
STFRD+PMML [9]	79.5 ±2.5	88.6	19.9
VSOF+OSS [23]	79.7 ±1.8	89.4	20.0
DeepFace-single	**91.4** ±1.1	96.3	8.6

Deep Learning Face Representation by Joint Identification-Verification

מאמר זה מציג רשת בשם DeepID2 המהווה שיפור לרשת DeepID שהוצגה בפרק הראשון. השיפור נעשה בעזרת פונקציית contrastive loss, כאשר פונקציה זו פועלת על שתי תמונות במקום על תמונה אחת. פונקציה זו מלמדת את הרשת להוציא ווקטורי תכונות באופן זה שווקטורי תכונות השייכים לאותו אדם יהיו קרובים אחד לשני, ובעוד שווקטורי תכונות של אנשים שונים יהיו רחוקים אחד מהשני. במילים אחרות, contrastive loss מקבלת ערך נמוך אם התמונות מקודדות באופן דומה, וערך גבוה אם התמונות מקודדות באופן שונה. מטרת הרשת היא למזער את הפער של תמונות השייכות לאותו אדם $(\mathbf{intra} - \text{class gap})$ ובמקביל למקסם את הפער בין תמונות השייכות לאנשים שונים$(\mathbf{inter} - \text{class gap})$. כפי שראינו בDeepID פונקציית softmax יכולה למקסם את הפער בין שתי מחלקות, אך עם זאת יש לה השפעה קטנה על הפער ה $\mathbf{intra} - \text{class}$. בשיטה זו נוסיף לsoftmax גם את פונקציית contrastive loss ובכך נמזער את הפער ה$\mathbf{intra} - \text{class}$, ובעזרת שילוב של שתי פונקציות אלו נלמד את ווקטור התכונות המייצג את תמונת הפנים. נקרא לפונקציה אחת סינגל הזיהוי, ולשנייה סיגנל האימות:

1. סיגנל זיהוי – סיווג תמונת קלט למחלקה במערך אימונים, נעשה שימוש ב-softmax .

2. סיגנל אימות – סיווג זוג תמונות כשייכות לאותה זהות או לא(סיווג בינארי), הפונקציה תוגדר במדויק במהלך הפרק.

בדומה לשיטה המוצעת בפרק הראשון גם בשיטה זו נלמד את תכונות התמונה על-ידי פירוקה לטלאים (patches) מאזורים שונים ובגדלים שונים, נחלץ מכל טלאי את התכונות שלו על-ידי DeepID2 ולאחר מכן נשרשר התכונות שחולצו עבור כל טלאי בשביל לקבל ייצוג סופי של תמונת הקלט. בדומה לפרק הראשון גם בשיטה זו מבנה הרשת DeepID2 יורכב ממספר רשתות בהתאם למספר הטלאים שהוצאו.

לאחר חילוץ התכונות של תמונת הפנים יהיה ניתן לאמן מסווג על בסיס אותם תכונות לצורך ביצוע אימות.

האלגוריתם המוצע השיג דיוק של 99.15%, שיפור של 1.7% מגרסתו הקודמת.

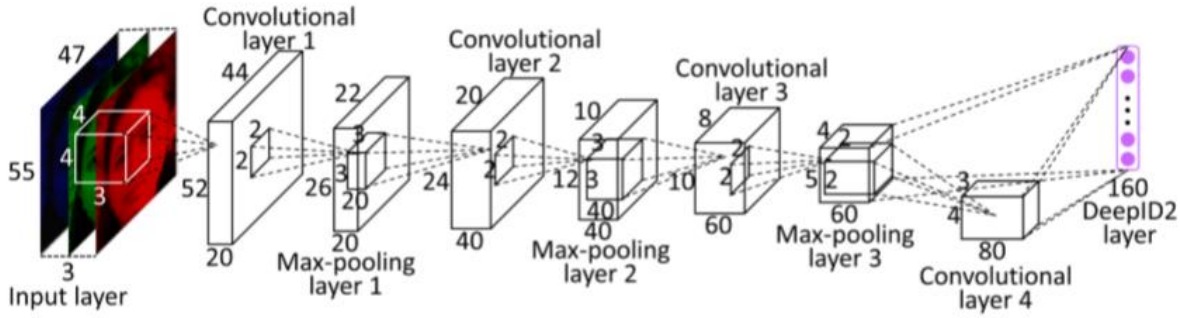
הסבר זה מהווה הצגה כללית של השיטה המוצעת במאמר, ובמהלך הפרק אסביר את השיטה בצורה יותר מעמיקה ואת תוצאות השיטה.

ארכיטקטורת הרשת תוכננה כך שקודם יחלצו תכונות בצורה היררכית, מתכונות מקומיות ברמה נמוכה לתכונות גלובליות ברמה גבוהה. הרשת מכילה ארבע שכבות קונבלציה כך שלאחר השלושה השכבות קונבלציה הראשונות יש שכבת $max - pooling$.

בשביל ללמוד מגוון רחב של תכונות ברמה גבוהה **לא** נעשה שימוש ב $weight - sharing$ בשכבות הקונבלציה העליונות, שכבת הקונבלציה השלישית שבה משקל כל ניורון משותף באופן **לוקאלי** בכל אזור בגודל 2×2 בתמונה. בשכבה קונבלציה הרביעית שהיא שכבה מחוברת לוקאלית אין שום משקל משותף בין נוירונים.

שכבת התכונות האחרונה שמכונה גם שכבת DeepID2 היא מימד 160 והיא מחוברת לחלוטין לשכבה השלישית **וגם** לשכבה הרביעית. מכיוון ששכבת הקונבלציה הרביעית מוציאה תכונות יותר גלובליות מאשר השכבה השלישית, שכבת DeepID2 תקבל כקלט multi − scale features. ומכיוון ש Rule מפיק תוצאות טובות יותר מאשר Sigmoid עבור מערכי נתונים גדולים, נעשה בו שימוש בשכבות השונות.

קלט הרשת מקבל תמונת RGB מימד 55×47.



התהליך של חילוץ התכונות מסומן ב- $f = Conv(x, \theta_c)$ כאשר:

- $Conv(\cdot)$ – הוא פונקציית הוצאת התכונות שמוגדרת להיות רשת קונבלציה.
- x – טלאי של תמונת פנים.
- θ_c – הפרמטרים של הרשת שיש ללמוד.
- f – ווקטור DeepID2 המחולץ.

ווקטור התכונות DeepID2 נלמד תחת שתי סיגנלים כאשר הראשון הוא סיגנל זיהוי, המסווג כל תמונת פנים לאחת מ n (למשל $n = 8192$) מחלקות שונות. סיגנל זה הוא למעשה שכבת $n - way\ softmax$ המונחת על שכבת הDeepID2 . כאשר מוציאה כפלט הסתברות מעל אותם

n מחלקות. הרשת אומנה בשביל למזער את הcross entropy loss שנכנה אותה "פונקציית הפסד הזיהוי" ונסמנה ב:

$$Ident(f,t,\theta_{id}) = -\sum_{i=1}^{n} -p_i \log \hat{p}_i = -\log \hat{p}_t$$

כאשר:

- $-f$ DeepID2 ווקטור.
- $-t$ תווית המחלקה(target class).
- $-\theta_{id}$ פרמטרים של שכבת softmax.
- $-p_i$ ווקטור המטרה כאשר $p_i = 0$ לכל i מלבד $p_t = 1$ עבור מחלקת המטרה t.
- $-\hat{p}_i$ פרדיקציה/חיזוי/פלט.

בשביל לסווג את כל המחלקות שכבת ה- DeepID2 צריכה ליצור תכונות הקשורות להבדל בין המחלקות השונות, כלומר תכונות עם וריאציות inter $class$ גדולות.

הסיגנל השני הוא סיגנל האימות שמטרתו היא לחלץ תכונות דומות כאשר מדובר בתמונות פנים של אותו אדם. כלומר במהלך האימון ובהינתן שתי תמונות פנים של אותו אדם המטרה היא להוציא ווקטורי DeepID2 שמרחקם קטן כמה שיותר אחד מן השני, ומרחקם יהיה גדול כאשר שתי תמונות הפנים יהיו שייכים לאדם שונה. מכיוון שיש כמה דרכים למדוד מרחק בין ווקטורים ניתן להגדיר את פונקציית ההפסד גם בכמה דרכים. **הדרך הראשונה** להגדיר את פונקציית ההפסד עבור סיגנל האימות היא בעזרת מרחק אוקולדי באופן הבא:

$$f(f_i, f_j, y_{ij}, \theta_{ve}) = \begin{cases} \frac{1}{2}\|f_i - f_j\|_2^2 & , \quad if\ y_{ij} = 1 \\ \frac{1}{2}\max(0, m - \|f_i - f_j\|_2)^2 & , \quad if\ y_{ij} = -1 \end{cases}$$

כאשר:

- $-f_i, f_j$ ווקטורי DeepID2 משתי תמונות שונות.
- $y_{ij} = 1$ – ציון לכך שf_i ו-f_j הם ווקטורי DeepID2 של **אותו** אדם. ובמקרה זה נמזער את המרחק $L2$ בין שתי הווקטורים.
- $y_{ij} = -1$ – ציון לכך ש f_i ו-f_j הם ווקטורי DeepID2 של אנשים **שונים**, ובמקרה זה נרצה שמרחקם יהיה גדול מ m.
- $\theta_{ve} = \{m\}$ – הפרמטר שצריך ללמוד.
- $\|x\|_2 = \sqrt{x_1^2 + \cdots + x_n^2}$

כלומר כאשר ווקטורי DeepID2 f_i ו-f_j שייכים לאנשים שונים נרצה שהמרחק ביניהם יהיה גדול מערך m . ואם זה אכן המצב אז הערך $\|f_i - f_j\|_2$ המציין את המרחק ביניהם יעלה על m מה שיגרור שהשגיאה של הפונקציה תהיה 0.

הדרך השנייה להגדיר את פונקציית ההפסד עבור סיגנל האימות היא בעזרת $L1\ norm$ והיא תהיה זהה לפונקציה שהוגדרה לפני כן מלבד כך שיעשה שימוש $\|x\|_1 = \sum_{i=1}^{n} |x_i|$

הדרך השלישית להגדיר את פונקציית ההפסד היא על-ידי $Cosine\ similarity$, שזו דרך למדוד דמיון בין שתי ווקטורים שלא שווים לאפס במרחב מכפלה פנימית.

$$f(f_i, f_j, y_{ij}, \theta_{ve}) = \frac{1}{2}\left(y_{ij} - \sigma(wd + b)\right)^2$$

כאשר: $\sigma(x) = \frac{1}{1+e^{-x}}$, $d = \frac{f_i \cdot f_j}{\|f_i\|_2 \|f_j\|_2}$

- $\theta_{ve} = \{w, b\}$ – הפרמטרים שצריך ללמוד.
- y_{ij}- מטרה בינארית של שתי תמונות הפנים המשוואות.

בהמשך תעשה השוואה בין שלושת פונקציות ההפסד.

המטרה היא ללמוד פרמטרים θ_c בפונקציית $Conv(\cdot)$ והפרמטרים מתעדכנים בעזרת $stochastic\ gradient\ descent$. הגרדינטים של הזיהוי והאימות משוקללים על-ידי פרמטר λ. לאחר אימון הרשת נוכל להוציא לכל תמונה ווקטור DeepID2 שישמש לאימות פנים.

input: training set $\chi = \{(x_i, l_i)\}$, initialized parameters θ_c, θ_{id}, and θ_{ve}, hyperparameter λ, learning rate $\eta(t)$, $t \leftarrow 0$

while not converge **do**

$t \leftarrow t + 1$ sample two training samples (x_i, l_i) and (x_j, l_j) from χ

$f_i = \text{Conv}(x_i, \theta_c)$ and $f_j = \text{Conv}(x_j, \theta_c)$

$\nabla\theta_{id} = \frac{\partial \text{Ident}(f_i, l_i, \theta_{id})}{\partial \theta_{id}} + \frac{\partial \text{Ident}(f_j, l_j, \theta_{id})}{\partial \theta_{id}}$

$\nabla\theta_{ve} = \lambda \cdot \frac{\partial \text{Verif}(f_i, f_j, y_{ij}, \theta_{ve})}{\partial \theta_{ve}}$, where $y_{ij} = 1$ if $l_i = l_j$, and $y_{ij} = -1$ otherwise.

$\nabla f_i = \frac{\partial \text{Ident}(f_i, l_i, \theta_{id})}{\partial f_i} + \lambda \cdot \frac{\partial \text{Verif}(f_i, f_j, y_{ij}, \theta_{ve})}{\partial f_i}$

$\nabla f_j = \frac{\partial \text{Ident}(f_j, l_j, \theta_{id})}{\partial f_j} + \lambda \cdot \frac{\partial \text{Verif}(f_i, f_j, y_{ij}, \theta_{ve})}{\partial f_j}$

$\nabla\theta_c = \nabla f_i \cdot \frac{\partial \text{Conv}(x_i, \theta_c)}{\partial \theta_c} + \nabla f_j \cdot \frac{\partial \text{Conv}(x_j, \theta_c)}{\partial \theta_c}$

update $\theta_{id} = \theta_{id} - \eta(t) \cdot \theta_{id}$, $\theta_{ve} = \theta_{ve} - \eta(t) \cdot \theta_{ve}$, and $\theta_c = \theta_c - \eta(t) \cdot \theta_c$.

end while

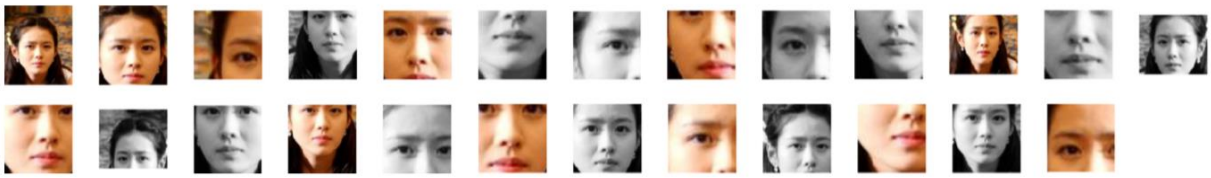
output θ_c

תהליך הוצאת התכונות עבור תמונת פנים עובד באופן הבא:

השלב הראשון - בהינתן תמונת קלט הוא לאתר 21 נקודות ציון (facial landmarks), הוצאת נקודות הציון נעשית על-ידי אלגוריתם SDM המתואר במאמר Supervised Descent Method and its Applications to Face Alignment. ולאחר מכן נעשה יישור של התמונה על-ידי טרנספורמציית דמיון על פי אותם הנקודות.

השלב השני - הוא בדומה לגרסה הקודמת של השיטה התמונה מחולקת ל-400 טלאים השונים במיקומם, במימדם ובמספר ערוצי הצבע שלהם וגם לכל תמונה נעשה סיבוב אופקי. הרשת עצמה בנויה ממספר רשתות שתלוי במספר הטלאים, כלומר רשת קונבלציה עבור כל טלאי. כל תת רשת כזו מקבלת טלאי לפני סיבוב אופקי ואחרי סיבוב, לכן נעשה שימוש ב-200 רשתות קונבלציה. בהתאמה לכך עבור כל תמונת קלט נקבל 400 ווקטורי DeepID2 כאשר כל ווקטור הוא מימד 160($400 \times 160 = 64000$).

בשלב השלישי - נעשה שימוש ב-$\text{forward} - \text{backward greedy}$ המתואר במאמר Greedy Algorithm for Sparse Learning with Linear Models בשביל לבחור מספר נמוך יותר של ווקטורי DeepID2 (בשיטה זו נבחרו 25), במטרה להשאיר את האלגוריתם פרקטי. באיור למטה ניתן לראות 25 טלאים שנבחרו בהתאם ווקטורי DeepID2 המרכיבות את ווקטור הייצוג מימד $25 \times 160 = 4000$.



השלב הרביעי - והאחרון הוא צמצום נוסף של הווקטור(ל180) על-ידי PCA, ועל סמך ווקטור זה נאמן מודל נוסף המבצע אימות פנים.

בשלב אימות הפנים אימנו את מודל Joint Bayesian על בסיס ווקטורי DeepID2 מהשלב הקודם.

בשביל לאמן את האלגוריתם השתמשו בערך הנתונים + CelebFaces (202599 תמונות פנים של 10177 מחלקות). ובשביל להעריך את התוצאות נעשה שימוש ב-LFW. **וחשוב להדגיש שבין המערכי נתונים אין כלל חפיפה.**

מתוך + CelebFaces נבחר באופן רנדומלי 8192 מחלקות בשביל לאמן את DeepID2 ותת מערך נתונים זה יכונה A + CelebFaces.

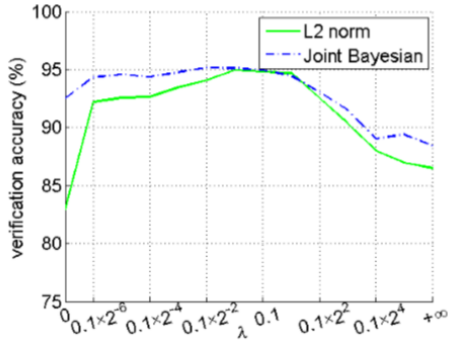
מ-1985 המחלקות הנותרות המכונות CelebFaces + B, הוצאו ווקטורי DeepID2 שישמשו בשביל לאמן את מודל Joint Bayesian לביצוע אימות פנים.

בניסויים הבאים נעשה שימוש ב$L2\ norm$ כסיגנל אימות לצורך אימון.

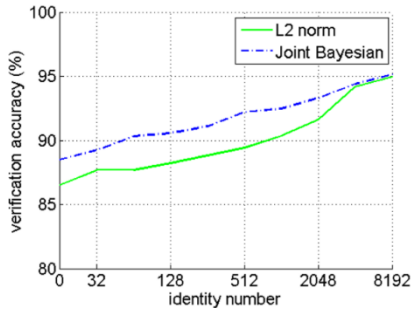
כפי שהוסבר קודם לכן האיזון של סיגנל הזיהוי וסיגנל האימות נקבע לפי פרמטר λ, ונבדק הקשר שבין פרמטר λ לביצועי המודל.

- כאשר $\lambda = 0$ אז סיגנל האימות מתאפס ורק סיגנל הזיהוי משפיע.
- כאשר λ גדל כך גם סיגנל האימות משפיע יותר.
- כאשר $\lambda \to +\infty$ כלומר רק סיגנל האימות משפיע.

האיור הבא מציג את הדיוק עבור בעיית האימות (test set) על-ידי השוואה של ווקטורי DeepID2 עם מרחק אוקולדי ושל Joint Bayesian, כתלות בפרמטר λ. וניתן לראות שהדיוק המרבי הושג כאשר היה שילוב בין שתי הסיגנלים. והדיוק הנמוך ביותר התקבל במקרי קצה $\lambda = 0$ ו $\lambda \to +\infty$.



בנוסף בדקו את ביצועי האלגוריתם כתלות במספר המחלקות עליו הוא אומן. בכל ניסוי העלו את מספר המחלקות בצורה אקספוננציאלית מ-32 מחלקות ועד 8192. ונראה קשר ישיר בין מספר המחלקות לבין דיוק האלגוריתם.



בשביל לבחון את סינגל האימות (L2) בוצעה השוואה על כל הזוגות החיובים(זוגות של תמונות שונות השייכות לאותו אדם) ועל כל הזוגות השליליים (זוגות של תמונות שונות השייכות לאנשים שונים). ונסמן ב- $L2 +$ ו- $L2 -$ בהתאמה. כלומר, ה- $L2 +$ רק מקטין את המרחקים בין DeepID2, ואילו $L2 -$ רק מגדיל את המרחקים בין DeepID2. (הדיוק עבור אימות הפנים שנלמד על בסיס ווקטורי DeepID2 , נמדד על-ידי Joint Bayesian וגם על-ידי המרחק האוקלידי בין הווקטורים). בנוסף לכך נבדקו ביצועים עבור $cosine$ ו-$norm$ L1 וגם ללא שימוש בסיגנל אימות כלל(none) . ובכל השוואות נעשה שימוש ב- 8192 מחלקות.

verification signal	L2	L2+	L2-	L1	cosine	none
L2 norm (%)	94.95	94.43	86.23	92.92	87.07	86.43
Joint Bayesian (%)	95.12	94.87	92.98	94.13	93.38	92.73

ניתן לראות שווקטורי DeepID2 שנלמדו עם $L2 +$ טיפה פחות טובים מL2 (כלומר צומצם המרחק בין DeepID2), ובניגוד לכך $L2 -$ נותן תוצאות דומות לתוצאות שלא נעשה כלל שימוש בסיגנל אימות(מגדיל את המרחק בין DeepID2) וזו ראייה לכך **שההשפעה של סיגנל האימות משפיע בעיקר על הפער $intra - class$ שזה בעצם מטרת הסיגנל.**

הבחנה נוספת היא שהדיוק הכללי עולה כאשר נעשה שימוש בסיגנל אימות. בנוסף לכך ניתן לראות שL2 כסיגנל אימות מפיק את התוצאות הגבהות ביותר מבין השאר ויתכן שזה נובע מכך שכל האילוצים האחרים חלשים יותר מ- L2 ופחות יעילים להפחתת הפער $intra - class$.

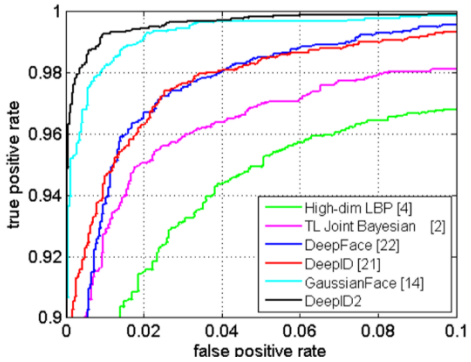
לפני אימון Joint Bayesian מבוצע PCA על ווקטורי DeepID2 לצמצום המימד ל 180 לאחר מכן מאמנים את Joint Bayesian על $CelebFaces + B$ ובודקים על 6000 זוגות של תמונות מ LFW. הטבלה למטה מסכמת את הזמן שלקח להוציא ווקטורי DeepID2 ואת הדיוק שהושג כתלות במספר הטלאים. הדיוק הגבוהה ביותר(98.43%) הושג בשימוש של 25 טלאים ווקטור תכונות מימד 180.

# patches	1	2	4	8	16	25
accuracy (%)	95.43	97.28	97.75	98.55	98.93	98.97
time (ms)	1.7	3.4	6.1	11	23	35

בשלב האחרון הפעילו את אלגוריתם הבחירה $forward - backward\ greedy$ שש פעמים נוספות, ובכל פעם בחרו DeepID2 (שהוצאו מטלאים) שלא נבחרו בשלב הקודם. לאחר מכן אימנו שוב את Joint Bayesian על אותם קבוצות ולבסוף הושג דיוק של 99.15%.

בטבלה ובאיור שלמטה ניתן לראות השוואה גרפית בין השיטה המוצעת לבין שיטות אחרות עד אותו מאמר.

method	accuracy (%)
high-dim LBP [4]	95.17 ± 1.13
TL Joint Bayesian [2]	96.33 ± 1.08
DeepFace [22]	97.35 ± 0.25
DeepID [21]	97.45 ± 0.26
GaussianFace [14]	98.52 ± 0.66
DeepID2	99.15 ± 0.13



לסיכום: תהליך העבודה שהוצג בשיטה זו דומה לתהליך העבודה של השיטה שקדמה לה, כלומר: הוצאת טלאים מתמונה, מבנה הרשת כולה שמורכב ממספר רשתות שתלוי במספר הטלאים ואחר מכן מודל נוסף המבצע אימות פנים, רק שבניגוד למאמר הקודם כאן לא הוצגה שיטה אחרת מלבד Joint Bayesian לביצוע אימות פנים בשלב האחרון. וההבדל העיקרי שהוצג בשיטה זו הוא השימוש בפונקציה נוספת בשביל להקטין את $intra - class$. בהמשך העבודה נראה שנעשה שימוש ברעיון זה אך בעזרת פונקציות שונות.

FaceNet: A Unified Embedding for Face Recognition and Clustering

במאמר זה דן במערכת בשם FaceNet, למערכת יש ארכיטקטורה ייחודית למשימות:

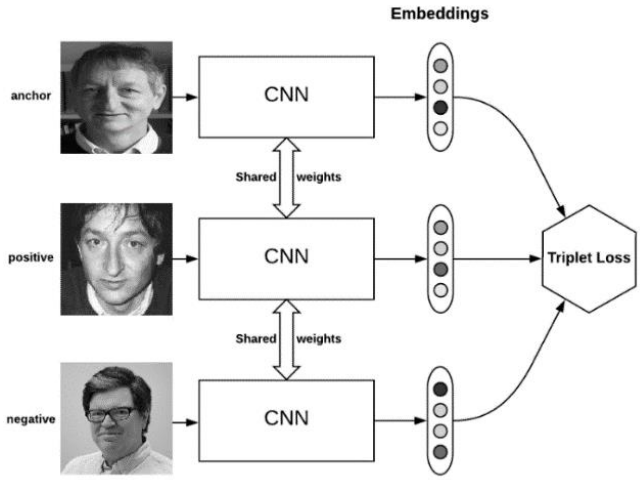
- אימות פנים – בהינתן שתי תמונות של פנים, לקבוע האם מדובר באותו אדם.
- זיהוי פנים – בהינתן תמונה של פנים, נזהה את הפנים מתוך מערך נתונים נתון.
- Clustering - מצא פנים דומים.

אציג **סקירה כללית** על FaceNet ולאחר מכן ארד לפרטים: FaceNet ממפה כל תמונת פנים למרחב אוקלידי ככה שהמרחק בתוך המרחב יהיה מקושר לדמיון בין תמונות פנים. כלומר תמונת פנים של בן אדם X תמוקם יותר קרוב לשאר התמונות של אותו אדם X במרחב בהשוואה לתמונות פנים של בן אדם Y.

לאחר שנקבל תמונה ונקודד אותה לווקטור תכונות ניתן לעשות אימות פנים, זיהוי פנים, ו Clustering בשיטות סטנדרטיות המוכרות מתחום הלימדת מכונה. למשל אפשר להשתמש ב-k-Nearest Neighbors algorithm לזיהוי פנים תוך כדי שימוש באותו בווקטור תכונות.

מערך האימון של FaceNet מורכב משלשות של תמונות: (1)תמונת עוגן(anchor) (2)תמונה חיובית, שזה תמונה של אותו אדם בתמונה בתמונת עוגן אך כמובן שהתמונה עצמה שונה. (3) תמונה שלילית, שזה תמונה של בן אדם שונה. בעזרת מערך אימונים מסוג זה נחשב את פונקציית ההפסד בשם "הפסד משולש" (triplet loss) . המודל עצמו הוא בצורת "רשתות סיאמיות" כלומר מורכב מכמה ענפים של רשתות קונבלציה כמוצג באיור, **והפרמטרים בכל הענפים זהים**.

אז לסיכום הסקירה **הכללית**, FaceNet עושה שימוש ברשתות קונבלציה והיא מאומנת כך שהמרחק האוקלידי בריבוע(L2) בין ווקטורי תכונות של פנים מקושר לדמיון של אותם פנים בתמונה. ולכן נרצה ללמוד פרמטרים של הרשת כך שנקבל קידוד טוב לתמונות הפנים. והרשת עצמה מאומנת על שלשות של תמונות כתואר לפני כן.



כמו שהוסבר בסקירה הכללית FaceNet עושה שימוש ברשת קונבלציה, והרשתות שנעשות בהם שימוש הם:

- The Zeiler&Fergus style networks
- Inception networks

הפרטים של רשתות אלו יפורטו בהמשך, ובינתיים לפישוט הדיון נתייחס למודלים אלו כקופסה שחורה. החלק הכי חשוב בשיטה מסתמך על כך שהמערכת כולה היא מערכת end-to-end, כלומר בעזרת המערך האימונים שלה ניתן לקבל פלט ללא הנדסה ידנית.

אנחנו רוצים ליצור וקטור תכונות $f(x) \in \mathbb{R}^d$ (embedding) מתמונת פנים נתונה x כך **שהמרחק האוקלידי בין כל תמונות הפנים של אותו בן אדם יהיה קטן, ואילו המרחק האוקלידי בין 2 תמונות שהם לא אותו אדם יהיה גדול.** אחת הדרכים ללמוד פרמטרים של הרשת שיתנו קידוד טוב לתמונות פנים זה להגדיר ולבצע Gradient descent על פונקציית הפסד משולש (triplet loss).

בשביל ללמוד את הפרמטרים של הרשת נצטרך להסתכל על שלוש תמונות באותו זמן.

1. תמונת עוגן x_i^a - תמונה של בן אדם מסוים.
2. תמונה חיובית x_i^p - תמונה של אותו אדם בתמונה בתמונת עוגן אך כמובן שהתמונה עצמה שונה.
3. תמונה שלילית x_i^n - תמונה של בן אדם שונה מהתמונת עוגן ומהתמונה החיובית.

ומה שנרצה זה שהמרחק בין תמונת העוגן לבין התמונה החיובית יהיה קטן, ואילו שהמרחק בין התמונת עוגן לתמונה השלילית יהיה גדול.

באופן פורמלי מה שנרצה זה:

$$\| x_i^a - x_i^p \|_2^2 + \alpha \leq \| x_i^a - x_i^n \|_2^2 \qquad \forall (x_i^a, x_i^p, x_i^n) \in \tau$$

כאשר τ הוא אוסף של שלשות מתוך המערך אימונים. ולכן נרצה ש:

$$\| f(x_i^a) - f(x_i^p) \|_2^2 + \alpha \leq \| f(x_i^a) - f(x_i^n) \|_2^2 \qquad \forall (x_i^a, x_i^p, x_i^n) \in \tau$$

ששקול ל-

$$|| f(x_i^a) - f(x_i^p) ||_2^2 - || f(x_i^a) - f(x_i^n) ||_2^2 + \alpha \leq 0$$

פרמטר α נקרא גם השוליים(margin) והוא מבטיח לנו שהרשת לא תלמד פתרון הטריוויאלי למשוואה. בלעדיו היינו מקבלים את האי-שיוון הבא:

$$|| f(x_i^a) - f(x_i^p) ||_2^2 - || f(x_i^a) - f(x_i^n) ||_2^2 \leq 0$$

כאשר הפתרון הטריוויאלי לאי שיוון זה הוא:

$$|| f(x_i^a) - f(x_i^p) ||_2^2 = || f(x_i^a) - f(x_i^n) ||_2^2$$

אז בסה"כ בהינתן שלוש תמונות (x_i^a, x_i^p, x_i^n) נגדיר את פונקציית ההפסד להיות:

$$f(x_i^a, x_i^p, x_i^n) = \max\{|| f(x_i^a) - f(x_i^p) ||_2^2 - || f(x_i^a) - f(x_i^n) ||_2^2 + \alpha, \ 0\}$$

האפקט של לקיחת המקסימום בין 0 לבין הביטוי הוא שאשר הביטוי משמאל קטן מ-0 אז השגיאה הכוללת תיהיה 0.

ולכן סה"כ עבור מערך אימונים בגודל N פונקציית ההפסד תיהיה:

$$\sum_i^N \left[|| f(x_i^a) - f(x_i^p) ||_2^2 - || f(x_i^a) - f(x_i^n) ||_2^2 + \alpha \right]_+$$

הבחירה של השלשות היא **בחירה קריטית**, וישנם **שלושה** אפשריות לבחירתם.
בהינתן תמונות פנים נרצה לבנות מערך אימונים שמורכב משלשות (x_i^a, x_i^p, x_i^n).

1) <u>שלשה קלה</u> – זו שלשה שמקיימת את המשוואה:

$$|| f(x_i^a) - f(x_i^p) ||_2^2 + \alpha \leq || f(x_i^a) - f(x_i^n) ||_2^2$$

כלומר המרחק בין תמונת העוגן לבין התמונה חיובית קטן מהמרחק בין תמונות העוגן לבין התמונה השלילית. ועל פי הגדרת פונקציית ההפסד הרשת לא באמת תלמד כי השגיאה תהיה אפס עבור הדוגמאות האלו.

2) <u>שלשה קשה</u> – זו שלשה שמקיימת:

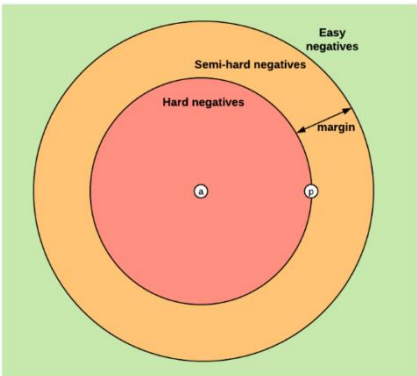
$$argmin_{x_i^n} || f(x_i^a) - f(x_i^n) ||_2^2 \ , \ argmax_{x_i^p} || f(x_i^a) - f(x_i^p) ||_2^2 .$$

כלומר שמהמרחק בין תמונת העוגן לתמונה החיובית יהיה גדול, והמרחק בין תמונה העוגן לתמונה השלילית יהיה קטן. גישה זאת עלולה להוביל למציאת מינימום **לוקאלי** בתחילת האימון.

3) <u>שלשה סמי-קשה</u> – זו שלשה שמקיימת את המשוואה:

$$\| f(x_i^a) - f(x_i^p) \|_2^2 < \| f(x_i^a) - f(x_i^n) \|_2^2 < \| f(x_i^a) - f(x_i^p) \|_2^2 + \alpha$$

כלומר שלשות ככה שהמרחק בין התמונה השלילית לתמונת עוגן גדול מהמרחק בין תמונת העוגן לבין התמונה החיובית, אבל שעדיין נקבל ביטוי חיובי בתוך פונקציית הפסד.



כותב המאמר כמובן בוחר להשתמש בשיטה הסמי-קשה. ולכן ברור שלא ניתן לעבור על מערך התמונות בצורה רנדומלית ופשוט ליצור שלשות כי זה יוביל אותנו לשלשות קלות. וגם לעבור על על מערך האימונים תמונה תמונה ולבדוק שלשות שמקיימות את התנאים לא ישים.

אז השאלה כעת היא איך לדגום את המערך בשביל לקבל שלשות כמו שתיארתי לפני כן, המאמר מציע שתי גישות:

- <u>Generate triplets offline</u> – בגישה זו נחשב לכל תמונה x במערך אימונים את $f(x)$ השייכת לה. כלומר "נכין" מראש לכל תמונה במערך אימונים את הווקטור תכונות שלה, ואז לפני כל epoch נבחר רק את השלשות הקשות או הסמי קשות . באופן כללי השיטה הזאת לא ממש יעילה מכיוון שנצטרך להעביר את כל מערך האימונים הנתון ברשת על מנת ליצור שלשה באופן שאנו רוצים.

- <u>Generate triplets online</u> - לחשב את $argmax$ ואת $argmin$ בתוך *mini-batch*. בשביל לקבל ייצוג מתאים למרחק שבין תמונות עוגן לתמונות חיוביות, צריך שיהיה מספר מינימלי של תמונות פנים השייכות לבן אדם מסוים בתוך *mini-batch* . כותב המאמר דגם את מערך האימונים ככה ש40 תמונות פנים (בקירוב) יהיו שייכים לאותו אדם בתוך ה-*mini-batch* . ובנוסף לצטרף ל-*mini-batch* תמונות פנים שליליות. וכעת במקום לבחור את התמונות החיוביות הקשות נבחר זוגות של תמונות חיוביות אך עדיין לבחור את התמונות השליליות הקשות.

בפועל שיטת ה-online הביאה את הרשת להתכנסות יותר מהירה בתחילת האימון, וזו השיטה שנבחרה למימוש המודל.בכל הניסויים הם אימנו את הרשת הקונבולציה בעזרת Stochastic GradientDescent ו-AdaGrad. וברוב הניסויים קצב הלמידה אותחל להיות 0.05 כאשר ערכו ירד במהלך האימון.

ובנוסף נבחר $\alpha(\text{margin}) = 0.2$.המודל אותחל בצורה רנדומלית בדומה ל-GoogLeNet שהוצג במאמר Going Deeper with Convolutions. ואומן על CPU כ1000 – 2000 שעות. קצב הירידה של פונקציית ההפסד ואיתו קצב העליית דיוק של המודל נהיה איטי בצורה משמעותית לאחר 500 שעות אימון.
כותב המאמר עשה שימוש בשתי ארכיטקטורות כאשר ההבדלים המעשיים שלהם הם הפרמטרים. טיב המודלים תלוי בשימוש, למשל מודל שרץ במרכז שרתים גדול יכול לכלול בתוכו הרבה פרמטרים ואילו מודל שרץ על פלאפון צריך להיות עם מספר נמוך של פרמטרים בשביל שיוכל להתאים לזיכרון של המכונה.

המודל הראשון מוסיף שכבות קונבולציה בגודל $1 \times 1 \times d$ (כמוצע במאמר Network In Network) בין שכבות הקונבולציה הסטנדרטיות בארכיטקטורה שמוצעת במאמר The Zeiler&Fergus style networks . כתוצאה מכך התקבל מודל עם 22 שכבות ו-140 מיליון פרמטרים.

layer	size-in	size-out	kernel	param	FLPS
conv1	220×220×3	110×110×64	7×7×3, 2	9K	115M
pool1	110×110×64	55×55×64	3×3×64, 2	0	
rnorm1	55×55×64	55×55×64		0	
conv2a	55×55×64	55×55×64	1×1×64, 1	4K	13M
conv2	55×55×64	55×55×192	3×3×64, 1	111K	335M
rnorm2	55×55×192	55×55×192		0	
pool2	55×55×192	28×28×192	3×3×192, 2	0	
conv3a	28×28×192	28×28×192	1×1×192, 1	37K	29M
conv3	28×28×192	28×28×384	3×3×192, 1	664K	521M
pool3	28×28×384	14×14×384	3×3×384, 2	0	
conv4a	14×14×384	14×14×384	1×1×384, 1	148K	29M
conv4	14×14×384	14×14×256	3×3×384, 1	885K	173M
conv5a	14×14×256	14×14×256	1×1×256, 1	66K	13M
conv5	14×14×256	14×14×256	3×3×256, 1	590K	116M
conv6a	14×14×256	14×14×256	1×1×256, 1	66K	13M
conv6	14×14×256	14×14×256	3×3×256, 1	590K	116M
pool4	14×14×256	7×7×256	3×3×256, 2	0	
concat	7×7×256	7×7×256		0	
fc1	7×7×256	1×32×128	maxout p=2	103M	103M
fc2	1×32×128	1×32×128	maxout p=2	34M	34M
fc7128	1×32×128	1×1×128		524K	0.5M
L2	1×1×128	1×1×128		0	
total				140M	1.6B

NN1 (Zeiler&Fergus)

המודל השני מבוסס על GoogLeNet style Inception models. השוני בין הארכיטקטורות של רשת זו לרשת רגילה, הוא שברשת רגילה אנחנו בוחרים את גודל הפילטרים בעוד ש Inception models נעשה שימוש במספר פילטרים בגדלים שונים בשכבה מסוימת באותו הזמן ושרשור של הפלטים באותה שכבה. למודל מסוג זה כמו שנעשה בו שימוש במאמר יש בין 6.6 – 7.5 מיליון פרמטרים, ולהלן הפירוט שלו:

type	output size	depth	#1×1	#3×3 reduce	#3×3	#5×5 reduce	#5×5	pool proj (p)	params	FLOPS
conv1 (7×7×3, 2)	112×112×64	1							9K	119M
max pool + norm	56×56×64	0						m 3×3, 2		
inception (2)	56×56×192	2		64	192				115K	360M
norm + max pool	28×28×192	0						m 3×3, 2		
inception (3a)	28×28×256	2	64	96	128	16	32	m, 32p	164K	128M
inception (3b)	28×28×320	2	64	96	128	32	64	L_2, 64p	228K	179M
inception (3c)	14×14×640	2	0	128	256,2	32	64,2	m 3×3,2	398K	108M
inception (4a)	14×14×640	2	256	96	192	32	64	L_2, 128p	545K	107M
inception (4b)	14×14×640	2	224	112	224	32	64	L_2, 128p	595K	117M
inception (4c)	14×14×640	2	192	128	256	32	64	L_2, 128p	654K	128M
inception (4d)	14×14×640	2	160	144	288	32	64	L_2, 128p	722K	142M
inception (4e)	7×7×1024	2	0	160	256,2	64	128,2	m 3×3,2	717K	56M
inception (5a)	7×7×1024	2	384	192	384	48	128	L_2, 128p	1.6M	78M
inception (5b)	7×7×1024	2	384	192	384	48	128	m, 128p	1.6M	78M
avg pool	1×1×1024	0								
fully conn	1×1×128	1							131K	0.1M
L2 normalization	1×1×128	0								
total									7.5M	1.6B

NN2 (Inception)

חלק ממודלים אלו הוקטנו בצורה דרסטית בגודל ככה שיוכלו לרוץ פלאפון.

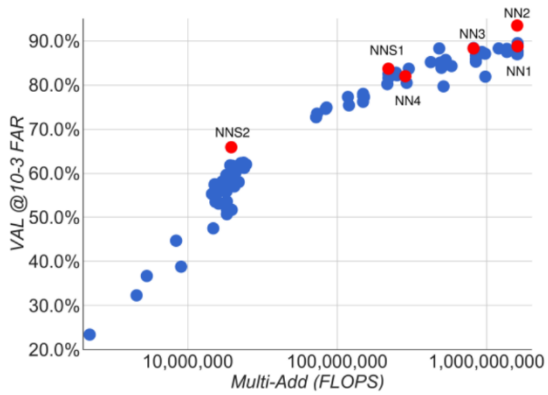
- (Zeiler&Fergus 220 × 220) **NN1** – 140 מיליון פרמטרים.
- (Inception 224×224) **NN2**.
- (mini Inception 165 × 165) **NNS1** – 26 מיליון פרמטרים.
- (tiny Inception 140 × 116) **NNS2** – 4.3 מילון פרמטרים.
- (Inception 160×160) **NN3**.
- (Inception 96 × 96) **NN4**.

הערכה של המודל נעשתה על ארבע מערכי נתונים שונים, כאשר הערכה בוצעה על **אימות פנים.** כלומר בהינתן זוג של תמונות פנים (i,j) נחשב את המרחק בין התמונות $\mathrm{D}(x_i, x_j)$ ונעשה שימוש בערך סף d בשביל לקבוע האם מדובר באותו בן-אדם בזוג תמונות (i,j).
נסמן את הזוגות (i,j) *של אותו בן אדם ב-* ρ_{same}. *ואת הזוגות השונים ב-* ρ_{diff}.
וכעת נגדיר את כל הזוגות שסווגו נכון ע"י: $TA(d) = \{(i,j) \in \rho_{same} \mid D(x_i, x_j) \leq d\}$
וכל הזוגות שסווגו לא נכון כאותו אדם ע"י: $FA(d) = \{(i,j) \in \rho_{diff} \mid D(x_i, x_j) \leq d\}$
ונגדיר את $VAL(d)$ validation rate ואת $FAR(d)$ false accept rate עבור ערך סף כלשהו להיות:

$$VAL(d) = \frac{|TA(d)|}{|\rho_{same}|}, FAR(d) = \frac{|FA(d)|}{|\rho_{diff}|}$$

כותבי המאמר הציגו trade-off שבין דיוק המודל לבין מספר ה- floating point operations per second (FLOPS) שבעצם מודד את מספר הפעולות שהרשת עושה.
והראו שיש קשר חזק בין מספר פעולות של המודל ובין הדיוק שלו.

אך trade-off בין הדיוק לבין מספר הפרמטרים של המודל לא היה ברור, למשל המודל NN2 הציג ביצועים דומים ל-NN1 כאשר יש לו מספר קטן בצורה משמעותית של פרמטרים.



סך-הכל הביצועים שכל מודל השיג הם:

architecture	VAL
NN1 (Zeiler&Fergus 220×220)	$87.9\% \pm 1.9$
NN2 (Inception 224×224)	$89.4\% \pm 1.6$
NN3 (Inception 160×160)	$88.3\% \pm 1.7$
NN4 (Inception 96×96)	$82.0\% \pm 2.3$
NNS1 (mini Inception 165×165)	$82.4\% \pm 2.4$
NNS2 (tiny Inception 140×116)	$51.9\% \pm 2.9$

המודל השיג ביצועים טובים על סוגים שונים של תמונות אבל הביצועים ירדו מעט כאשר היה מדובר בתמונות קטנות יותר בגדלים 120X120 ועבור 80X80 , מה שסה"כ הגיוני כי הרשת אומנה על תמונות בגודל 220X220. וככל הנראה שאימון על תמונות באיכות נמוכה היו פותרים את הבעיה.

jpeg q	val-rate
10	67.3%
20	81.4%
30	83.9%
50	85.5%
70	86.1%
90	86.5%

#pixels	val-rate
1,600	37.8%
6,400	79.5%
14,400	84.5%
25,600	85.7%
65,536	86.4%

#dims	VAL
64	$86.8\% \pm 1.7$
128	$87.9\% \pm 1.9$
256	$87.7\% \pm 1.9$
512	$85.6\% \pm 2.0$

גם בחנו את מימד הווקטור שמופק מהרשת, היה הגיוני לחשוב שמימד וקטור גדול יהיה טוב לפחות כמו מימד וקטור ממנו. שימוש בווקטור תכונות ממימד 256 נתן ביצועים טיפה פחות טובים מווקטור מימד 128 וביצעים קצת יותר טובים מווקטור ממימד 512. בסופו של דבר נבחר להפיק וקטור תכונות באורך 128.

ההערכות בוצעו על בסיסי נתונים הבאים:

- LFW - מכיל יותר מ13 אלף תמונות של פנים שנאספו מהאינטרנט וכל תמונה מתויגת לפי שם של הבן אדם בתמונה. 1680 אנשים שונים שיש להם שתיים או יותר תמונות. על מערך נתונים זה הושג דיוק של 99.87% .

- YouTube Faces Database - מכיל 3425 סרטונים של 1595 אנשים שונים כאשר מספר הפריימים הממוצע לסרטון הוא 181.3 . מעוצב בדומה לLFW . על מערך נתונים זה הושג דיוק של 95.12%
- Personal Photos – מכיל תמונות של שלושה אנשים שביחד מגיעים ל12 אלף תמונות. לא היה רשום דיוק על מערך נתונים זה.

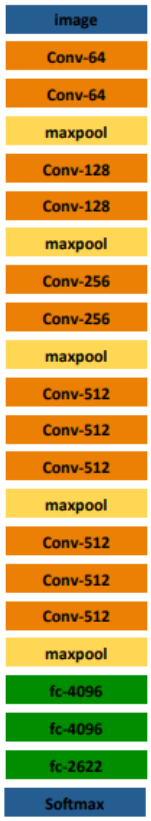
לסיכום: המאמר הראה שיטה לקידוד תמונות פנים לתוך מרחב אוקלידי בצורה ישירה וזה בשונה מעבודות אחרות שעשו שימוש ב-PCA וכמו כן גם בסיווג בעזרת SVM.
השיטה עובדת מקצה לקצה ובכך מפשטת את השיטה עצמה. וגם בניגוד לעבודות אחרות המודל לא דורש עיבוד מקדים על התמונה מלבד חיתוך של הפנים מתוך התמונה.

Deep Face Recognition

- **"Very Deep" Architecture**
  - Different from previous architectures proposed for face recognition:
    - locally connected layers (Facebook)
    - inception (Google)
- **Learning a multi-way classifier**
  - Soft Max Objective
  - 2622 way classification
  - 4096d descriptor
- **Learning Task Specific Embedding**
  - The embedding is learnt by minimizing the triplet loss

$$\sum_{(a,p,n)\in T} \max\{0, \alpha - \|\mathbf{x}_a - \mathbf{x}_n\|_2^2 + \|\mathbf{x}_a - \mathbf{x}_p\|_2^2\}$$

- Learning a projection layer from 4096 to 1024 dimensions
- Online triplet formation at the beginning of each iteration
- Fine tuned on target datasets



בניגוד למאמרים מהפרקים הקודמים מאמר זה נכתב באקדמיה באוניברסיטת אוקספורד וללא קשר לתעשייה. הסיבה העיקרית שראיתי נכון להדגיש את העובדה הזאת היא בגלל ההבדל המשמעותי במשאבים. למשל במאמר *FaceNet* שבפרק הקודם נעשה שימוש במערך נתונים המכיל **200 מיליון תמונות**. כותב המאמר מדגיש את העובדה שאין לקבוצות מחקר באקדמיה את היכולת לבנות מערכי אימונים ענקיים כאלו. ולכן במאמר זה יש שני דיונים מרכזיים.

1. תהליך המסביר את **בניית מערך הנתונים**, ואני כבר אציין שהמערך מכיל סך-הכל **2.6 מיליון תמונות**.
2. הצגת ארכיטקטורת הרשת VGG Face ואימונה.

אימון הרשת היא שילוב של הרעיונות המרכזיים של המאמרים הקודמים ומתוארת בתהליך דו-שלבי:

השלב הראשון הוא לאמן את הרשת לסווג תמונות פנים בעזרת softmax כאשר יש 2622 מחלקות. **השלב השני** הוא להשתמש ברשת שאומנה לסווג, להסיר את השכבה שמבצעת את הסיווג ואז לאמן שכבה נוספת בעזרת triplet loss**.** לאחר אימון הרשת כולה נוכל לתת לרשת כקלט שתי תמונות פנים ולקבל את הייצוג שלהם במרחב האוקלידי ולאחר מכן לבצע השוואה ביניהם בעזרת מרחק אוקולדי.

ארכיטקטורת הרשת VGG Face יחסית פשוטה ביחס לארכיטקטורות של המאמרים הקודמים וגם מערך הנתונים שנעשה בו שימוש, אבל הביצועים של המערכת כולה טובים מה שלפי דעתי מדגיש את החשיבות בשימוש של מערך נתונים איכותי.

כמו שנאמר לפני כן, החלק הראשון הוא איסוף נתונים ובניית מערך נתונים ככה שהמערך נתונים יכיל אלפי זהויות (אנשים) שלכל אחד מהם יהיו מאות תמונות. איסוף הנתונים מבוצע בתהליך רב-שלבי.

שלב הראשון- יצירת רשימה ראשונית של מועמדים:

השלב הראשון הוא יצירת רשימת שמות של מועמדים שיהיה אפשר להשתמש בתמונות שלהם. הרעיון הוא להתרכז בשמות של אנשים מפורסמים ודמיות ציבוריות ככה שיהיה קל למצוא מגוון רחב של תמונות שלהם ברחבי האינטרנט. אתחול הרשימה נעשתה מתוך *Internet Movie Data Base* כאשר מתוך מאגר נתונים זה הוצאה רשימה של גברים ונשים המדורגים לפי פופולריות. לאחר מכן מבוצעת הצלבה של רשימה זו עם כל האנשים ב- *Freebase knowledge graph* כאשר מערך אימונים זה מכיל מידע 500 אלף אנשים. לאחר ההצלבה נותרנו עם רשימה המכילה 2500 גברים ו2500 נשים מפורסמים ככה שיש לנו מידע לגבי כל אחד מהמועמדים כמו מוצא, גיל וכדומה.

לאחר מכן רשימת מועמדים זו עברה סינון כאשר סוננו מועמדים שאין עבורם מספיק תמונות באינטרנט. לאחר פעולה זו הורדו 200 תמונות עבור כל זהות מתוך הרשימה של ה-5000 זהויות(בעזרת מנוע חיפוש של גוגל עבור תמונות) .

לאחר מכן 200 התמונות שמחלוקות לארבע בלוקים שונים כאשר בכל בלוק יש 50 תמונות הוצגו לאנשים לצורך פיקוח על טיב התמונות. אותם אנשים התבקשו לשמור על זהות מסוימת רק אם לפחות 90% מ-200 התמונות שהוצגו להם היו איכותיות. פעולה זו הפחיתה את רשימת המועמדים ל-3250 זהויות ברשימה.

לאחר מכן ביצעו הצלבה נוספת עם מערכי הנתונים LFW ו- *YTF* והסרו כל המועמדים שחופפים לאותם מערכי נתונים וזה במטרה שיהיה אפשר לאמן את המודל על מערך נתונים חדש ולהעריך את התוצאות בצורה אמינה על אותם מערכי נתונים. בסופו של דבר נותרה רשימה של 2622 זהויות.

שלב שני- איסוף תמונות נוספות עבור כל זהות:

לכל זהות מתוך הרשימה נעשה חיפוש של תמונות בעזרת מנועי חיפוש שונים, וכתוצאה מכך מערך הנתונים עמד על 2000 תמונות לכל זהות.

שלב שלישי- שיפור מערך הנתונים בצורה אוטומטית להשלים:

מטרת שלב זה היא להסיר את כל הפרצופים השגויים בכל קבוצה בצורה אוטומטית באמצעות סיווג ולשם כך נעשה שימוש ב-*SVM*. עבור כל זהות נבחרו 50 התמונות הכי "מוצלחות"

ששימשו כדוגמאות חיוביות, ו-50 תמונות הכי "מוצלחות" של כל זהות אחרת ששימשו כדוגמאות שליליות(כלומר נוצר סיווג בינארי של one-vs-rest) עבור כל זהות אימנו את המודל ובאמצעותו דרגו את כל 2000 התמונות של אותה זהות, והמשיכו עם ה1000 הטובות ביותר.

שלב רביעי- מחיקת תמונות כפולות

הוצאת תמונות כפולות שהתקבלו ממנועי חיפוש שונים, או תמונות כפולות שהתקבלו מאתרים שונים. גם Near duplicates (שזה למשל יכול להיות אותם תמונות עם ערכי צבע שונים) מוסרים גם כן. פעולות אלו מבוצעות בעזרת חישוב של VLAD descriptor עבור כל תמונה, וביצוע clustering על אותם מתארים כאשר **נבחר ערך סף קטן**.

שלב חמישי ואחרון – סינון ידני

בשלב זה יש 2622 זהויות ויותר מ-1000 תמונות לכל זהות. מטרת שלב זה היא להגדיל את איכות מערך הנתונים בעזרת סינון ידני, ובשביל להאיץ את זמן הסינון הידני נעשה שימוש גם ברשת קונבלציה. אומנה רשת AlexNet בשביל לבצע הבחנה בין 2622 הזהויות השונות ונעשה שימוש ב $softmax$ בשביל לתת ציון לכל תמונה. לאחר מכן הוצגו לאנשים התמונות עם הציונים שלהם בבלוקים 200 והם התבקשו לאשר בלוק בשלמותו.

ניתן לראות את סיכום השלבים, וכן השוואה למערכי נתונים מוכרים אחרים בטבלאות הבאות:

Stage	Aim	Type	# of persons	# of images per person	total # images	Annotation effort	100%-EER
1	Candidate list generation	A	5,000	200	1,000,000	–	–
2	Image set expansion	M	2,622	2,000	5,244,000	4 days	–
3	Rank image sets	A	2,622	1,000	2,622,000	–	96.90
4	Near dup. removal	A	2,622	623	1,635,159	–	–
5	Final manual filtering	M	2,622	375	982,803	10 days	92.83

Dataset	Identities	Images
LFW	5,749	13,233
WDRef [4]	2,995	99,773
CelebFaces [25]	10,177	202,599

Dataset	Identities	Images
Ours	2,622	2.6M
FaceBook [29]	4,030	4.4M
Google [17]	8M	200M

נסמן את ארכיטקטורת הרשת ב- $\emptyset$, ואת מספר המחלקות בה ב-N

אתחול הרשת כסיווג של התמונות פנים – בהמשך לדיון על מבנה מערך הנתונים נקבע $N = 2622$. הרשת קונבלציה תשייך לכל תמונה במערך נתונים $t = 1, \dots \dots, T$,l_t וקטור ציון($Score$)

$$x_t = W\emptyset(l_t) + b \ \in \mathbb{R}^N \ ; \quad W \in \mathbb{R}^{N\times D}, \qquad b \ \in \mathbb{R}^N$$

באמצעות שכבה מחוברת לחלוטין המכילה N ניורוני פלט, נירון לכל זהות במערך נתונים.
לאחר מכן ווקטור הציון מושווה עם מחלקת ה-ground-truth של אותה זהות $c_t \in \{1, \ldots\ldots, N\}$ וזה על-ידי חישוב:

$$softmax\ log - loss\ E(\emptyset) = -\sum_t \log\left(\frac{e^{\langle e_{c_t}, x_t\rangle}}{\sum_{q=1,\ldots.N} e^{\langle e_{c_t}, x_t\rangle}}\right)$$

כאשר $x_t = \emptyset(l_t) \in \mathbb{R}^D$ ו- e_c מציין את ה- one-hot vector של מחלקה c.

לאחר האימון של הרשת נסיר את שכבת הסיווג(W, b) כך שנשאר עם הווקטור$\emptyset(l_t)$. ואז ניתן לבצע אימות בין שתי ווקטורים על-ידי חישוב המרחק האוקלידי ביניהם. **אבל** ניתן לשפר את הדיוק של המודל כולו בכך שנבצע את תהליך האימות במרחב האוקלידי בעזרת triplet loss שהוסבר בפרק הקודם.

אתחול הרשת כמסווג כפי שהוסבר נמצא כדי להפוך את האימון לקל ומהיר באופן משמעותי.

כתזכורת לפרק הקודם מטרת אימון triplet loss הוא לבצע אימות על-ידי השוואה בין תיאורי פנים במרחק האוקלידי. ניקח את הפלט $\emptyset(l_t) \in \mathbb{R}^D$ של הרשת שאומנה מראש בשביל לבצע סיווג, ננרמל אותה ונטיל אותה על מימד $L \ll D$ על-ידי טרנספורמציה אפינית

$$x_t = \frac{W'\emptyset(l_t)}{\|\emptyset(l_t)\|_2} \quad W' \in \mathbb{R}^{L\times D}$$

יש שתי הבדלים משמעותיים בין האימון הקודם לאימון זה. הראשון הוא $L = 1024 \neq D$, כלומר L לא שווה למספר המחלקות במערך נתונים. ההבדל השני הוא שההטלה W' אומנה בשביל למזער את הtriplet loss

$$E(W') = \sum_{(a,p,n)\in T} \max\{0,\ \alpha - \ ||x_a \ - x_n\ ||_2^2 \ + \ |||x_a \ - x_p\ ||_2^2 \quad x_i = W'\frac{\emptyset(l_t)}{\|\emptyset(l_t)\|_2}$$

$\alpha \geq 0$ representing a learning margin

T is a collection of training triplets

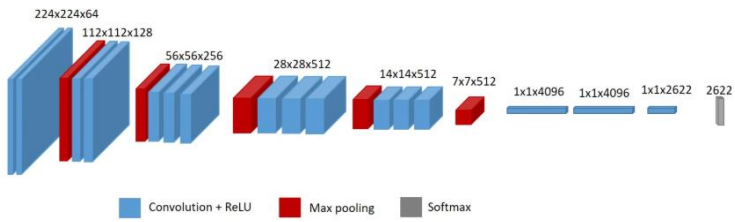
הם הביאו בחשבון שלוש ארכיטקטורות שונות A, B, D .

- **ארכיטקטורה של A**: מכילה 22 שכבות:
  - 16 שכבות קונבלציה.
  - 5 שכבות max pooling.
  - שכבת $softmax$.

התאור **המלא** של הרשת כפי שהוצג במאמר, ואיור ויזואלי של הרשת.

layer	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
type	input	conv	relu	conv	relu	mpool	conv	relu	conv	relu	mpool	conv	relu	conv	relu	conv	relu	mpool	conv
name	–	conv1_1	relu1_1	conv1_2	relu1_2	pool1	conv2_1	relu2_1	conv2_2	relu2_2	pool2	conv3_1	relu3_1	conv3_2	relu3_2	conv3_3	relu3_3	pool3	conv4_1
support	–	3	1	3	1	2	3	1	3	1	2	3	1	3	1	3	1	2	3
filt dim	–	3	–	64	–	–	64	–	128	–	–	128	–	256	–	256	–	–	256
num filts	–	64	–	64	–	–	128	–	128	–	–	256	–	256	–	256	–	–	512
stride	–	1	1	1	1	2	1	1	1	1	2	1	1	1	1	1	1	2	1
pad	–	1	0	1	0	0	1	0	1	0	0	1	0	1	0	1	0	0	1

layer	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37
type	relu	conv	relu	conv	relu	mpool	conv	relu	conv	relu	conv	relu	mpool	conv	relu	conv	relu	conv	softmx
name	relu4_1	conv4_2	relu4_2	conv4_3	relu4_3	pool4	conv5_1	relu5_1	conv5_2	relu5_2	conv5_3	relu5_3	pool5	fc6	relu6	fc7	relu7	fc8	prob
support	1	3	1	3	1	2	3	1	3	1	3	1	2	7	1	1	1	1	1
filt dim	–	512	–	512	–	–	512	–	512	–	512	–	–	512	–	4096	–	4096	–
num filts	–	512	–	512	–	–	512	–	512	–	512	–	–	4096	–	4096	–	2622	–
stride	1	1	1	1	1	2	1	1	1	1	1	1	2	1	1	1	1	1	1
pad	0	1	0	1	0	0	1	0	1	0	1	0	0	0	0	0	0	0	0



לאחר כל שכבת קונבלציה יש פעולת $Relu$ ושכבות max pooling לאחר שתיים או שלוש שכבות קונבלציה. השלוש שכבות קונבלציה האחרונות נקראות Fully Connected, שהם כמו שכבות קונבלציה אבל נעשה שימוש ב-1×1 convolution כאשר העומק של כל פילטר הוא כעומק תמונת הקלט. פירטתי על פעולה זו ועל הקשר שלה לFully Connected בנספח לפרק.

הפלט של שתי השכבות הראשונות שמחוברות לחלוטין הוא כמספר הפילטרים בהם, כלומר 4096. והפלט של השכבה האחרונה היא בהתאם לפונקציית הפסד שנעשה בה שימוש, כלומר $N = 2622$ או $L = 1024$ כאשר במקרה ש$N = 2622$ נעביר את הווקטור שהתקבל לשכבת softmax.

- **ארכיטקטורה של B:** זהה לארכיטקטורה של A אבל כוללת שתי שכבות קונבלציה נוספות.
- **ארכיטקטורה של D:** זהה לארכיטקטורה של A אבל כוללת **חמש** שכבות קונבלציה נוספות.

אימון הרשת כולה מתחיל קודם באימון הרשת שמבצעת סיווג. מטרת האימון היא למצוא פרמטרים של הרשת שממזערים את $softmax\ log - loss$.

אימון רשת A:

- **אופטימיזציה -** $stochastic\ gradient\ descent\ with\ momentum$ כאשר מקדם המומנטום הוא 0.9.
- **רגולציה –** לאחר שתי השכבות המחוברת לחלוטין נוספה שכבת $\text{dropout} = 0.5$.
- **קצב למידה –** אותחל להיות 10^{-2} ופחת בפקטור של 10 כאשר הדיוק על סט האימות הפסיק לעלות. ובמהלך האימון ערך זה פחת שלוש פעמיים.

- **אתחול משקולות –** בצורה רנדומלית מהתפלגות גאוסינית, וכל ערכי ה- $bias = 0$.
- **גודל הקלט –** כל תמונה בגודל 224×224.
- **אוגמנטציה –** היפוך התמונות משמאל לימין בהסתברות של 50%

אימון רשתות B, D :

הרשת A אומנה מאפס בעוד שרשתות B, D אומנו על הרשת **המאומנת** A. זה נעשה על-ידי הוספה של שכבות קונבלציה נוספות (שתיים וחמש בהתאמה) אם אתחול רנדומלי וקביעת קצב למידה קטן יותר(לא ציינו את הערך) ואז אימון של הרשת שוב.

בשביל ללמוד את הייצוג במרחב האוקלידי בעזרת triplet loss, הרשת "הקופאה" למעט השכבה האחרונה שמחוברת לחלוטין המיישמת את ההטלה. (הוקפאה - הכוונה שמשקלי הרשת לא ישתנו באותם שכבות שהוקפאו). **כלומר נלמד את השכבה שמבצעת הטלה ממימד 4096 למימד 1024.** שכבה זו נלמדת במשך 10 epochs ובעזרת *stochastic gradient descent* עם קצב למידה של 0.25. בשלב זה אנחנו מאמנים triplet loss ולכן צריך להתחשב בסידור מערך הנתונים בהתאם. כלומר סידור התמונות לשלשות (a, p, n) כאשר a זו תמונת העוגן, p זו תמונה חיובית(תמונה שונה של אותו אדם a) ו- p שזו תמונה שלילת השייכת לאדם שונה מ - a. בדומה לפרק 2 גם כאן הבחירה של השלשה קריטית ויש לה קשר ישיר לביצועי המודל. השיקולים לבחירת השלשה הוסברו בהרחבה בפרק הקודם ולכן אני לא רואה טעם לחזור עליהם שוב.

בחינת המודל לביצוע אימות - בהינתן הזהויות l_1, l_2 נרצה לדעת האם מדובר באותה זהות. אז "נוציא" מהרשת את הייצוגים שנלמדו $W'\emptyset(l_1)$, $W'\emptyset(l_2)$ ונבחן את המרחק ביניהם על-ידי:

$$\|W'\emptyset(l_1) - W'\emptyset(l_2)\|_2 < \tau \; ; \; \tau \; is \; some \; \text{threshold}$$

ערך τ לא נלמד במהלך האימונים שצוינו עד כה.

בשביל לעמוד את ביצועי המודל ולבצע השוואה לעבודות אחרות נעשה שימוש בשתי מערכי אימונים נוספים:

1. **LFW** מכיל 13323 תמונות מהאינטרנט של 5749 אנשים מפורסמים.
2. **YTF** מכיל 3425 סרטונים של 1595 אנשים מפורסמים. והמטרה של מערך אימונים זה הוא למדוד ביצועים של אימות פנים ברמת סרטונים. איכות התמונות במערך זה נמוכות יותר ממערכי האימונים אחרים כתוצאה מטשטוש בגלל תנועה או המרחק של הפנים בסרטון.

הערה: מערך הנתונים שנעשה בו שימוש לאימון הרשת מכיל גם הוא תמונות מפורסמים, אבל בזמן הבנייה שלו היה שלב שמסיר תמונות של אנשים שנמצאים ב LFW או ב YTF .

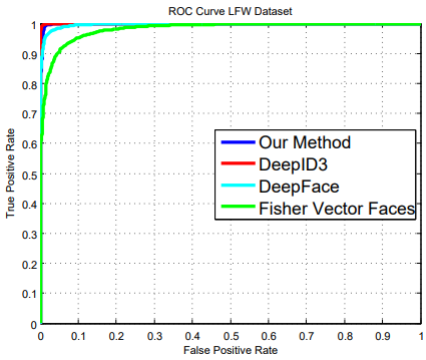
לפני השוואות ביצועים נציין שהאיתור פנים נילקח מהמאמר:
Face Detection without Bells and Whistles

ולצורך יישור תמונת פנים היה צריך לחשב נקודות ציון בתמונה וזה נעשה בשיטה שמתוארת ב-
Taking the bite out of automated naming of characters in TV video ולאחר מכן יושמה טרנספורמציית דמיון דו-מימדית בשביל למפות פנים לצורה קנונית.

ניתוח ביצועים של כל רכיב ברשת כאשר סט האימות הוא LFW.

- **יישור תמונת פנים**: יישור תמונות פנים במערך אימון אינה שיפרה את ביצועי המודל אבל יישור תמונות פנים על מערך **הבדיקה** כן שיפרה את הביצועים.
- **ארכיטקטורת הרשת:** ביצועי רשת B עלו על רשת A אבל מנגד רשת D הורידה את הביצועים. הסיבה לכך נעוצה בהוספה של חמש שכבות קונבלציה נוספות על רשת A ושלוש שכבות נוספות על רשת B, כלומר ל-D יש מספר רב יותר של פרמטרים.
- **Triplet-loss**: שיפר את ביצועי הרשת.

השוואה למודלים אחרים: הושגו תוצאות טובות ביחס לשיטות אחרות, וזה כאשר נעשה שימוש בהרבה פחות נתונים ורשת פחות מורכבת.



No.	Method	Images	Networks	Acc.
1	Fisher Vector Faces [27]	-	-	93.10
2	DeepFace [29]	4M	3	97.35
3	Fusion [30]	500M	5	98.37
4	DeepID-2,3		200	99.47
5	FaceNet [17]	200M	1	98.87
6	FaceNet [17] + Alignment	200M	1	99.63
7	Ours	2.6M	1	98.95

ניתן לראות את הבוצעים גם ביחס למודלים שהוסברו בפרקים הקודמים.

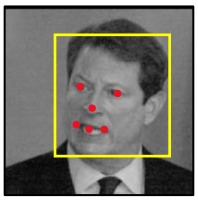
לסיכום: המאמר הציג כמה שיטות, שיטה אחת לבניית מערך נתונים איכותי ושיטה שנייה ויחסית פשוטה לביצוע אימות בין תמונות. הרעיון המרכזי באימון המודל הוא שילוב של שתי הרעיונות המרכזיים שהוצגו בDeepFace ו- FaceNet. נעשה שימוש גם ב softmax בהקשר של סיווג תמונות פנים שהוצגה במאמר DeepFace וגם נעשה שימוש ב-Triplet-loss שהיה הדיון מרכזי ב-FaceNet . אבל ההבדל המשמעותי הוא הארכיטקטורות הרשת ; נזכיר שב-DeepFace ארכיטקטורת הרשת תוארה בפירוט וב-FaceNet הארכיטקטורה אמנם לא תוארה לעומק אבל זה כי ארכיטקטורה הייתה מוכרת לפני כן אבל זה לא אומר שהיא פשוטה, ובטח יותר מורכבת מזו שהוצגה במאמר זה. השימוש ברשת פשוטה לפי דעתי רק מדגיש את החשיבות במערך נתונים איכותי

נספחים

תהליך יישור בדו-מימד

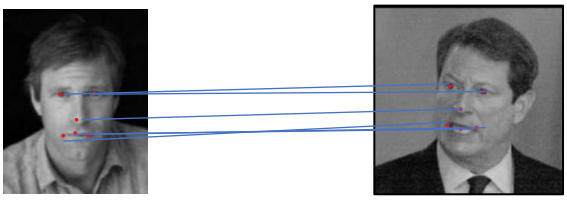
(חזרה לפרק 1, חזרה לפרק 2, חזרה לפרק 3)

בהינתן תמונות פנים, הדבר הראשון לעשות זה לחפש שש נקודות עוגן באופן שהוסבר לפני כן.



לאחר מכן ניקח תמונה אחרת שהיא כבר בפוזה קנונית, ונוציא ממנה גם שש נקודות עוגן.

ולאחר מכן נבצע התאמה בין אותם נקודות על מנת למצוא טרנספורמציה בין אותם פוזות.



ובעזרת טרנספורמציה נבצע יישור לתמונה שאנחנו רוצים:

טרנספורמציה אפינית

(חזרה לפרק 1)

היא פונקציה בין מרחבים אפינים אשר משמרת נקודות, קווים ישרים ומישורים. כמו כן, קבוצות של קווים מקבילים נשארים מקבילים לאחר טרנספורמציה אפינית. טרנספורמציה אפינית לא בהכרח שומרת על זוויות בין ישרים או מרחק בין נקודות אבל היא משמרת יחס מרחק בין נקודות שנמצאות על אותו ישר. טרנספורמציה אפינית היא ההרכבה של שתי פונקציות: הזזה והעתקה ליניארית. וניתן לייצג מיפוי זה על-ידי כפל וחיבור מטריצות, ובצורה פורמלית יותר:

אם ההעתקה הליניארית מיוצגת ככפל במטריצה A , וההזזה מיוצגת כחיבור וקטור $\vec{b}$, ניתן לייצג העתקה אפינית f הפועלת על וקטור $\vec{x}$ באופן הבא: $f(\vec{x}) = A\vec{x} + \vec{b}$

Local Binary Pattern **(LBP)**

(חזרה לפרק 2)

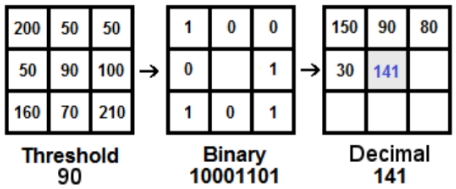
אלגוריתם יעיל אשר מתייג את הפיקסלים של תמונה על-ידי שימוש בערך סף שתלוי בשכניו של כל פיקסל בתמונה ומציג את התוצאה בצורה בינרית. שיטה זו הוכחה להיות יעילה עבור משימות סיווג. ובנוסף נמצא ששילוב של LBP ביחד עם היסטוגרמת גרדיינטים (HOG) משפר את הביצועים של הסיווג. על-ידי שילוב של השתיים ניתן להציג תמונת פנים בעזרת וקטור פשוט. ישנם ארבע פרמטרים שצריך להתחשב בהם באלגוריתם זה:

1. רדיוס – בעזרת הרדיוס ניתן לבנות מעגל לוקאלי לרוב מאותחל ב- 1.
2. שכנים – מספר השכנים שצריך להתחשב בהם בעת בניית המעגל, לרוב מאותחל ב-8.
3. X – מספר התאים בכיוון אופקי.
4. Y- מספר התאים בכיוון אנכי.

בנוסף צריך להחזיק מערך אימון לצורך אימון האלגוריתם, מערך האימונים יהיה מורכב מתמונות פנים של אנשים שאותם אנחנו רוצים לזהות כולל תיוג של אותה תמונה.

השלב הראשון באלגו' זה הוא ליצור תמונת ביניים שמתארת את התמונה בצורה טובה יותר על-ידי הדגשת מאפיינים בפנים. וזה נעשה בעזרת חלון הזזה על התמונה בהתבסס על הפרמטר **רדיוס** שציון לפני כן. נקבע את הערך שנמצא באמצע החלון בעזרת שכניו באופן הבא:

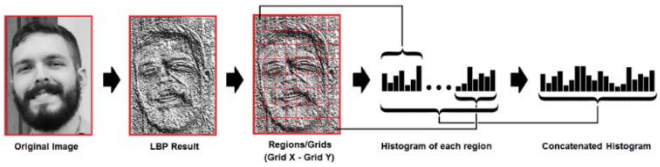
- עבור כל שכן של אמצע החלון, אם ערך של שכן זה גדול מערך של אמצע החלון תחליף את ערך השכן להיות 1, אחרת אם הוא קטן מאמצע החלון תשנה את הערך ל-0. כלומר התא במרכז החלון משתמש כערך סף בינרי.
- כעת לערך שבאמצע החלון יש שכנים שהם 0 או 1. נשרשר את ערכים אלו בשביל לקבוע את ערך אמצע החלון.



תהליך זה נקרא LBP, וכעת נשתמש בתמונה החדשה שנוצרה לנו ובפרמטרים Y,X בשביל לחלק את התמונה לחלונות קטנים יותר ועבור כל חלון שנוצר נוציא היסטוגרמה באופן הבא:

- כל היסטוגרמה מכל חלון שנוצר תכיל 256 תאים בהתאם לגווני אפור של התמונה.
- נשרשר את כל ההיסטוגרמות שנוצרו בשביל לקבל היסטוגרמה גדולה. למשל, נניח שהתמונה מחולקת להיות 8X8 חלונות אז יהיה לנו 16384 תאים עבור ההיסטוגרמה הגדולה.

וההיסטוגרמה הגדולה תהיה ייצוג של תמונת הפנים.



שילוש דלוני

(חזרה לפרק 2)

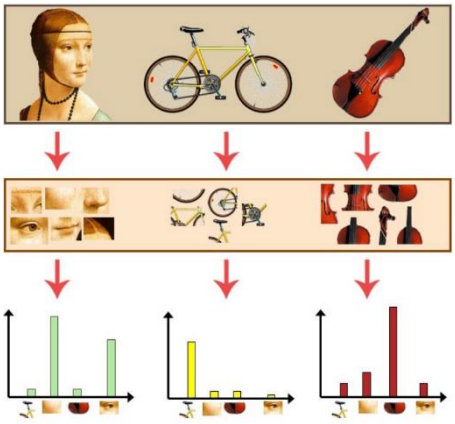
שילוש דלוני של קבוצת נקודות במישור הוא חלוקה של המישור למשולשים של קבוצת הנקודות כך שאף נקודה אינה נמצאת בתוך המעגל החוסם אחד מהמשולשים. תכונה זו של השילוש הופכת אותו לאופטימלי, מבחינות מסוימות, משום שהיא מבטיחה שהמשולשים שירכיבו את השילוש יהיו עבים ושמנים, ולא ארוכים ודקים.

bag of visual words/VLAD

(חזרה לפרק 5)

ובקיצור BOVW, הרעיון הכללי של ***bag of visual words*** הוא לייצג תמונה כאוסף של תכונות, כאשר התכונות מכילות נקודות עניין (*keypoints*) והמתארים של הנקודות(*descriptors*). נקודות עניין אלו נקודות בתמונה ככה שלא משנה אם התמונה סובבה, הוקטנה או הוגדלה, נקודות אלו לא ישתנו. והמתארים הם התיאור של נקודות העניין. משתמשים בנקודות העניין והמתארים שלהם בשביל לבנות "מילון" ולייצג כל תמונה כהיסטוגרמת התדיריות של התכונות. לאחר מכן מתוך היסטוגרמה זו יהיה ניתן למצוא דמיון בין תמונות.

בניית המילון נעשית על-ידי "הוצאת" תכונות לכל תמונה, והתכונות שהוצאו ישמשו כמילים במילון. ואז בהינתן אוסף של מילים, עבור כל תמונה ניתן לבנות היסטוגרמה של אותם מילים. כלומר נבדוק כמה מילים יש בתמונה ואת כמותם. למשל באיור המצורף בהיסטוגרמה האמצעית יש יותר מהמילים שנראות כמו חלקים של אופניים מאשר כל מילה אחרת.



לאחר הוצאת היסטוגרמה לכל תמונה נרצה לבצע השוואה. ניתן לבצע השוואה על-ידי מכפלה פנימית של שתי תמונות. ניקח את ההיסטוגרמות של התמונות, ננרמל אותם ל-1 ואז ניתן לחשוב על היסטוגרמה כווקטור מנורמל ולאחר מכן לבדוק את המכפלה הפנימית שלהם.

באופן פורמלי יותר, נניח שיש היסטוגרמה של תמונה במערך נתונים שנסמן אותה ב-$\overrightarrow{d_j}$ ויש לנו תמונה שהיא שאילתה שתסומן ב- $\vec{q}$ ננרמל את ההיסטוגרמות שלהם ונבצע השוואה שמוגדרת על-ידי:

$$sim(d_j, q) = \frac{\langle d_j, q \rangle}{\|d_j\| \, \|d_j\|} = \frac{\sum_{i=1}^{V} d_j(i) * q(i)}{\sqrt{\sum_{i=1}^{V} d_j(i)^2}} * \frac{1}{\sqrt{\sum_{i=1}^{V} q(i)^2}}$$

$$V \text{ is the number of words}$$

VLAD היא גם שיטת ייצוג תכונות של תמונות ומהווה הרחבה - *bag of visual words*.

1×1 convolution

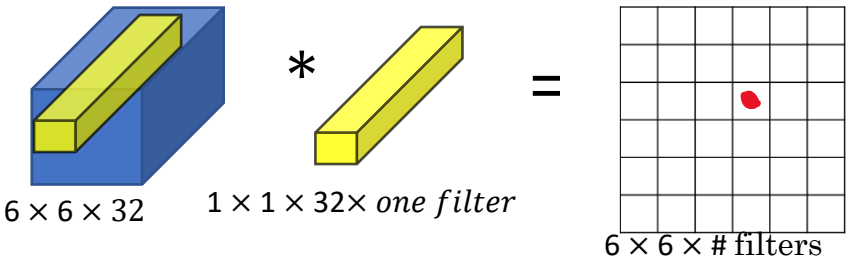
(חזרה לפרק 5)

אני אסביר את הפעולה עם דוגמא קונקרטית. נניח שתמונת הקלט היא מממד $6\times 6 \times 32$ ***אז פעולת קונבלציה ממד*** $1 \times 1 \times 32$ ***תסתכל על כל אחד מ36 המיקומים השונים בתמונת הקלט ותחשב את המכפלה הסקלרית ביניהם, ואז לעשות שימוש בפונקציית אקטיבציה.***

ניתן לחשוב על הפילטר אחד כזה כמו על ניורון יחיד שלוקח כקלט 32 מספרים מכפיל אותם 32 משקולות ולאחר מכן מיישם ***פונקציית אקטיבציה ומוציא כקלט את המספר שהתקבל.***

אז אם נניח שיש כמה פילטרים כאלו בשכבה ניתן לחשוב על שכבה זו כ-Fully Connected .

וגם באופן כללי ניתן להשתמש בפעולה זו בשביל להקטין את הממד של התמונה, הרי שהאורך הרוחב של תמונת הקלט לא ישתנה בפעולה זו, אבל מימד העומק שלהם ישתנה בהתאם למספר הפילטרים באותה שכבה.



$multi - scale\ features$

(חזרה לפרק 1, חזרה לפרק 3)

הפעולה נועדה בשביל לשלב תכונות משכבות שונות של הרשת בשביל למקסם את הפרדיקציה. למשל נניח שיש רשת שבה עשר שכבות קונבלציה, וניקח פלט מכל שכבה שנייה נשנה את הממדים של כל פלט כזה לאותו גודל ונשרשר אותם ביחד. וכעת נוכל לעשות שימוש במידע שיש לנו משכבות שונות ברשת בשביל לבצע פרדיקציה.

נרצה לעשות שימוש בפעולה זו בגלל שבכל שכבה נוספת ברשת אנחנו מקבלים תכונות יותר גלובליות, ולכן תכונות ברמה נמוכה יותר חסרות. ובמילים אחרות אם נסתכל על תכונה לוקאלית כלשהי בתמונה נתקשה לבצע חיזוי אבל ביחד עם הוספה של תכונה גלובלית זה נעשה יותר פשוט.

softmax and cross entropy loss

(חזרה לפרק 3)

אני אסביר את הפונקציה ביחד עם דוגמא:
נניח שווקטור **המטרה** הוא $p = (0,1,0,0)^T$ ונניח שהפלט שקיבלנו הוא $\hat{p} = (0.3,0.2,0.1,0.4)^T$
ונשתמש בפונקציית הפסד הבאה:

$$-\sum_{i=1}^{4} -p_i \log \widehat{p_i}$$

נשים לב שמתקיים בווקטור המטרה ש: $p_1 = p_3 = p_4 = 0$ ולכן מהפונקציה הפסד נקבל:

$$f(\hat{p}, p) = -\sum_{i=1}^{4} -p_i \log \widehat{p_i} = -p_2 \log \widehat{p_2} = -\log \widehat{p_2}$$

אז זה אומר שאם תהליך הלמידה רוצה למזער את פונקציית ההפסד $f(\hat{p}, p)$ הוא צריך למזער את $-\log \widehat{p_2}$ והדרך היחידה לעשות את זה היא להגיע לערכים גדולים כמה שאפשר עבור $\widehat{p_2}$

שכבות מחברות לוקאלית

(חזרה לפרק 1)

בשכבת קונבלציה רגילה אנחנו מריצים את אותו פילטר עבור כל מיקום (x, y) בתמונה. כלומר כל הפיקסלים בתמונה **חולקים** את אותו פילטר. בעוד ששכבה מחוברת לוקאלית עושה שימוש בפילטרים שונים עבור כל מיקום (x, y) בתמונה, כלומר כל פילטר לומד לזהות תכונה אחרת בתמונה. לדוגמא: נניח שאנו רוצים לאמן 256 פילטרים בגודל $(5x5)$ במקום מסוים ברשת. ונניח שהתמונה בגודל$(128x128)$ ולכן ניתן להכניס לתמונה מספר פילטרים בגודל $(124x124)$. מספר הפרמטרים עבור דוגמא זו יהיה .5 x 5 x 256 x 124 x 124 וזה לעומת מספר הפרמטרים בשכבת קונבלציה רגילה שהיא 5 x 5 x 256.

פעולה זו אכן עושה שימוש במספר רב יותר של פרמטרים, אך עם זאת בגלל שכל פיקסל מקבל פילטר משלו זה מצמצם את מספר הפילטרים הכולל שצריך לאמן ברשת.