פרויקט גמר : ספר פרוייקט

העשרת דיבור במיקרופון יחיד באמצעות רשת עמוקה מרובת מוצאים

מספר פרויקט403 :

מגיש :

ניר סגל

מנחה:

יוחאי ימיני

אחראי אקדמי:

פרופ' שרון גנות

תוכן עניינים

הבעיה שאותה אנו פותרים 4
מבוא ללמידה עמוקה 5
רשת נוירונים עמוקה 5
רקע על המבנה של רשת נוירונים העמוקה 6
השכבה השמאלית (input layer) 6
השכבות האמצעיות 6
השכבה הימנית (output layer) 6
שיטות תכנון רשת נוירונים 7
Perceptron 8
אלגוריתם העברת החישובים הרשת (Forward passing) 8
פונקציות אקטיבציה 9
סיגמוייד 9
ReLU 10
אלגוריתמים לצורכי אימון הרשת 11
Backpropagation and Gradient Descent 11
Batch 11
Gradient Descent and Backpropagation 11
Epoch 12
Learning Rate 12
Adam Optimizer 12
סט אימון 13
סט ולידציה (קרוס-ולידציה) 14
סט מבחן 14
Fully connected and MLP 14
רקע תאורטי על עיבוד האות בפרוייקט 15
STFT – Short Time Fourier Transform 15
רקע תאורטי על העשרת ספקטרום 16
Spectrogram Enhancement (Mapping Based) 18
הפתרון הממומש במאמר 19
רקע על שכבות שונות ברשתות נוירונים עמוקות 21
Fully Connected Layer 21
Dropout layer 21
אימון הרשת 22
פונקציית הפסד (Loss Function) 22
ניהול ה dataset 22
Training set – סט האימון 22
Validation set – סט הולידציה 22

Test set – סט המבחן 22
הגדרת הבעיה 23
עיבוד המידע 24
מימוש סינון ועיבוד של סט המידע 25
בניית הרשת..... 26
תוצאות 28
PESQ – Perceptual Evaluation of Speech Quality..... 28
STOI – Short Time Objective Intelligibility..... 28
דוגמא של שערוך עבור רעש מסוג Wideband :..... 30
דוגמא של שערוך עבור רעש מסוג Narrowband : 31
מסקנות ודוגמאות32
דוגמא ראשונה..... 32
דוגמא שנייה..... 32
דוגמא שלישית 32
דוגמא רביעית..... 32

בעולם בו הממשק בין האדם ובין הטכנולוגיה פורץ גבולות, החל מלחיצת כפתור ועד זיהוי קולי אשר נמצא בשימוש יום יומי בחיינו. קיים צורך ביכולות סינון, הבנה ועיבוד קול של אדם אל מול רעשי הסביבה הרבים. כאשר אדם ברחוב מבצע שיחה בטלפון הנייד שלו, הצד השני שומע את האדם היטב וללא רעשי רקע. דבר זה נובע מהעשרת דיבור, כלומר ניקוי האות מרעשים חיצוניים על ידי עיבוד האות, נסביר:
אות הדיבור שאנו מפיקים, כפוף לרעשי רקע הקיימים בעולם: כביש סואן, רוחות, ריבוי אנשים וכדומה... על ידי שימוש בעיבוד אותות ואמצעים כגון למידה עמוקה, ניתן לנקות את אות הדיבור מרעשי הסביבה, ובכך לקיים שיחה באופן בו שני הצדדים יכולים להבין את הנאמר ללא רעש.
העשרת דיבור הינה פונקציה חשובה ושכיחה מאוד בתחום עיבוד אותות.

בשנים האחרונות החלה להתפתח גישה חדשה ללמידת האות והרעשים לשם העשרה. גישה זו נקראת למידה עמוקה וממומשת בעיקר בתחום מדעי המחשב ועיבוד אותות.
הרעיון העומד מאחורי מערכת למידה עמוקה, הינו יכולת ההבנה, החשיבה וההסתגלות האוטומטית של המערכת. כל זאת נעשה על סמך אוסף של קבצי אימון שהמערכת לומדת וממשת את הדרך בה ניתן להעשיר את האות בצורה הטובה ביותר.

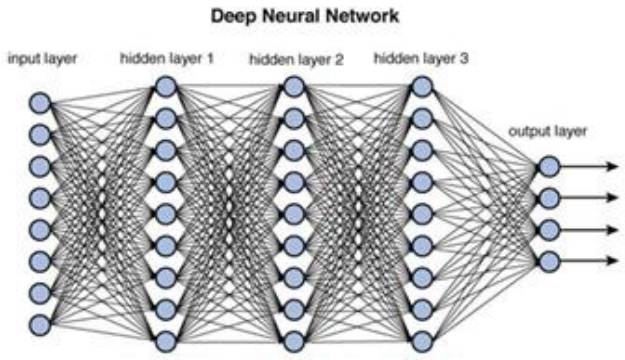
על מנת לשפר במידה ניכרת את מערכת הלמידה העמוקה, אנו נשלב בפרויקט זה 3 שיטות שונות להעשרת האות. כאשר המערכת תשכלל את שלושת השיטות בצורה המביאה למינימום את המרחק בין הספקטרוגרמה המשוערכת לנקייה.

מבוא ללמידה עמוקה

את הפרויקט נממש באמצעות עולם הלמידה העמוקה, לכן נציג את הרעיונות, האלגוריתמים ומושגי היסוד הרלוונטיים לפרויקט זה.

רשת נוירונים עמוקה

רשת נוירונים הינה מערכת למידה מפוקחת הבנויה ממספר גדול של אלמנטים פשוטים, המכונים נוירונים כאשר כל נוירון מסוגל לבצע בחירה של החלטה פשוטה ואת ההחלטה הזו הוא מעביר קדימה לנוירונים הבאים, כאשר הנוירונים מסודרים בצורה של שכבות אשר מחוברות זו לזו.
באמצעות הרשת, יהיה ניתן להביע כמעט כל פונקציה, ולכן ניתן לענות על משימות כמו מיון, סיווג וחישובים מסוגים שונים.
על מנת למטב את ביצועי הרשת נדאג לספק סט אימון רחב הכולל דוגמאות שונות של אותה בעיה אשר אנו מנסים לפתור.
המחשה של רשת נוירונים עמוקה המורכבת משכבות של נוירונים, כניסה ויציאה ושכבות נסתרות ונוירונים .



המחשה של רשת נוירונים עמוקה המורכבת משכבות של נוירונים, כניסה ויציאה ושכבות נסתרות ונוירונים .

רקע על המבנה של רשת נוירונים העמוקה

השכבה השמאלית (input layer)

מתארת את שכבת הקלט – בפרוייקט זה, לשכבה זו המשתמש יכניס את אות הדיבור הרועש.

השכבות האמצעיות

נקראות hidden layers, אלו הן שכבות הביניים, כאשר מספר שכבות אלו איננו מוגבל. בשכבות אלו מתבצע עיבוד ביניים לטובת חישוב השערוך, כל שכבה מכילה משקולות אשר מחושבות באמצעות גרדיאנטים, הנוירונים עוברים דרך פונקציית אקטיבציה לפני המעבר לשכבה הבאה.

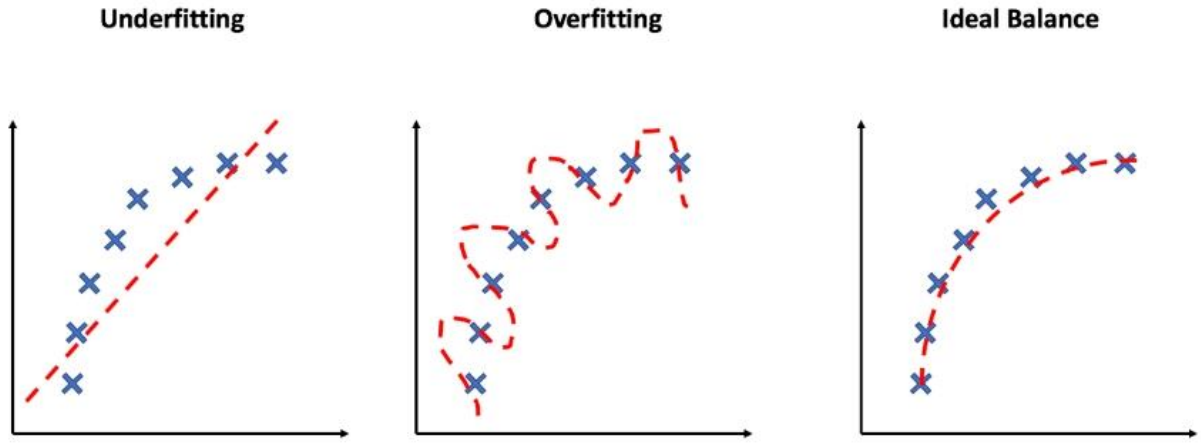
השכבה הימנית (output layer)

שכבת המוצא, אשר מקבלת את המידע שהועבר אליה דרך השכבות הנסתרות ומציגה חיזוי סופי. בפרוייקט זה, שכבה זו תייצג עבורנו שערוך של האות הנקי מרעש.

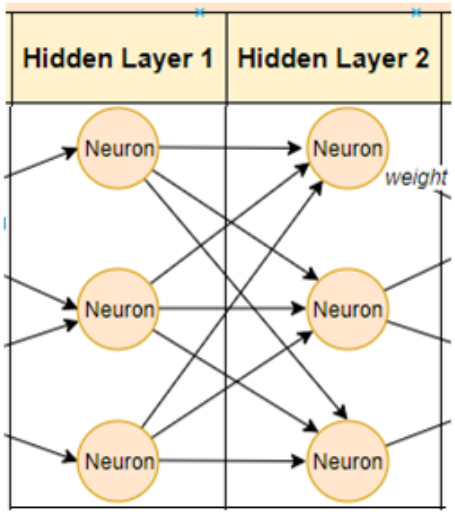
מבנה הרשת בפרוייקט יכיל שכבת כניסה, שלוש שכבות נסתרות וארבעה מוצאים של הרשת. כל שכבה נסתרת תהיה בגודל של 1024, שכבת הכניסה בגודל של Feature Vector , במקרה שלי, בגודל 1085, וכל מוצא יהיה בגודל של פריים רגיל מתוך הSTFT – כלומר, 257.

שיטות תכנון רשת נוירונים

הבעיה המרכזית ביותר בתכנון מערכות אלו היא בחירה של ההיפר פרמטרים, אלו הפרמטרים שאנו קובעים אשר עבורם הרשת תתן את הביצועים הטובים ביותר.
על מנת לבחור את ההיפר-פרמטרים, משתמשים בשיטה של קרוס ולידציה, כאשר מחלקים את הסט של האימון לסט של אימון וגם סט של קרוס-ולידציה (בדרך כלל נבחר את האחוז הדומיננטי של הדאטה סט לטובת האימון), את סט האימון נאמן דרך חישוב הגרדיאנטים ואת סט הקרוס ולידציה נעביר דרך הרשת ללא חישוב גרדיאנטים אך נשתמש בחישוב של השגיאה וכך נדע האם המערכת מצליחה לשערך מידע שהוא זר מהסט אימון, במילים אחרות אנחנו פותרים כך את בעיית הoverfitting שעלולה להגרם לרשת, במצב של overfitting אנחנו נראה ביצועים טובים של הרשת כלפי סט האימון אך לא כלפי מידע זר.



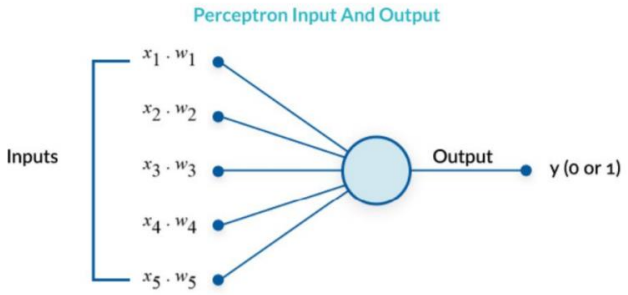
המחשה של overfitting, במצב זה ניתן לראות כי הקו עובר דרך הנקודות של סט האימון אך בין הנקודות קיימת אוסילציה.



המחשה של מבנה השכבות הנסתרות והחיבורים ביניהן

Perceptron

מודל שתפקידו לחקות את הנוירון במוח האנושי, ברשתות נוירונים נשתמש במודל זה על מנת לקבל החלטות ולהעבירן הלאה.
המושג perceptron הוא מושג בסיס בעולם רשתות הנוירונים והוא המרכיב המרכזי בשכבות של הרשת, באמצעותו ניתן יהיה לבצע חישובים (החלטות) ולהעבירן הלאה ברשת.

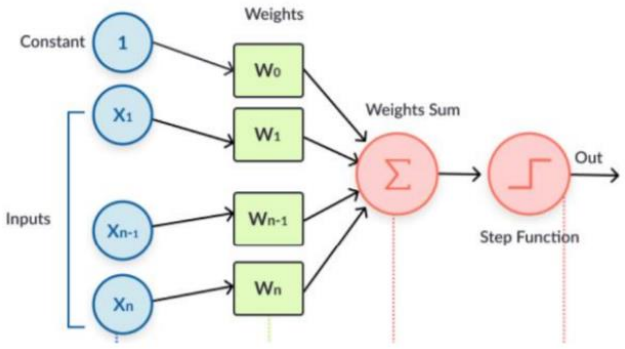


מבנה של perceptron

אלגוריתם העברת החישובים הרשת (Forward passing)

נתאר את התהליך שהמידע עובר בתור הרשת ודרך הנוירונים.

1. הכניסות המוזנות לרשת מוכפלות במשקולות.
2. סכימה של הכניסות המוכפלות במשקולות.
3. הוספה של קבוע bias על מנת לבצע הזזה לכיוון השערוך.
4. העברה של הסכום דרך פונקציית אקטיבציה.
5. העברת התוצאה הכוללת ככניסה של הנוירון הבא ברשת.



המחשה של אופן חישוב המידע ברשת נוירונים

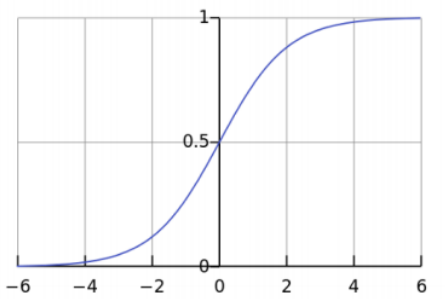
פונקציות אקטיבציה

שכבות הביניים מורכבות משרשור של שכבות נוירונים וביניהן פונקציות לא ליניאריות בשם פונקציות אקטיבציה. הנוירון בכל שכבה, הינו צירוף ליניארי של כל הנוירונים בשכבה הקודמת ועליו מופעלת פונקציית האקטיבציה. מטרת פונקציות האקטיבציה הינה הגדלת תחום הפונקציות אותן הרשת יכולה להביע. ללא פונקציות האקטיבציה, הרשת הייתה יכולה להביע רק פונקציות ליניאריות.
פונקציות אקטיבציה עיקריות שבהן נשתמש בפרוייקט זה:

סיגמוייד

הגדרת הפונקציה: $\sigma(x) = \frac{1}{1+e^{-x}}$

פונקציה זו מקבלת כך ערך ומנרמלת אותו לטווח מספרים בין 0 ל1.
נראה חסרון בשימוש פונקציה זו עבור ערכים גדולים, כיוון שבערכים אלו לא נראה שינוי בנגזרת, אלא הנגזרת תתאפס. דבר שכזה יקשה על האימון ולימוד המערכת. חסרון נוסף הינו שהנגזרת אי-שלילית, לכן מאפשרת רק גרדיאנט חיובי.
פונקציה זו איננה נמצאת בדרך כלל בשכבות הביניים, השימוש הנפוץ של פונקציה זו הוא בבעיות מסוג סיווג, לדוגמא, במערכת שתפקידה להחליט האם קלט של תמונה כלשהי מכיל חתול, סיגמוייד יציג לנו שערוך סופי בטווח של 0 עד 1 וכך נדע את הסיכוי של המצאות החתול בתוך התמונה (למשל, עבור ערך יותר קרוב ל-1, יש יותר סיכוי שקיים חתול בתמונה).
בפרויקט זה, השימוש בסיגמוייד יבוא לידי ביטוי כאשר נרצה לשערך את המסכות IBM ו-IRM, כאשר אלו נתונות מראש בטווח ערכים שבין 0 ל1

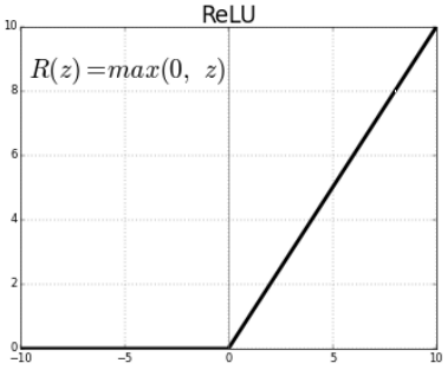


סרטוט של פונקציית הסימוייד, מקבלת כל ערך ומוציאה ערך בין 0 ל1.

ReLU

הגדרת הפונקציה: $R(z) = \max(0, z)$

ניתן לראות כי פונקציה זו איננה לינארית ומכאן יתרון השימוש שלה כפונקציה אקטיבציה ברשתות נוירונים, הפונקציה מאפסת כל ערך שלילי שהיא מקבלת ומעבירה הלאה כל ערך חיובי שהיא מקבלת.
לפונקציה זו שימוש נפוץ בעולם של רשתות נוירונים מאחר והיא זולה לשימוש כתוצאה מכך שישנם טווח ערכים שהיא מאפסת, ולכן חלק מהנוירונים לא יעבדו בו זמנית וכך נחסוך זמני חישוב.



סרטוט של פונקציית הReLU.

אלגוריתמים לצורכי אימון הרשת

Backpropagation and Gradient Descent

על מנת לבצע אופטימיזציה לרשת, שמטרתה לשערך בצורה כמה שיותר מדוייקת את האות ללא הרעש, נגדיר פונקציית שגיאה (Loss) על ידי חישוב ההפרש בין תוצאת הרשת ($\hat{y}$ - השיערוך) לבין האות הנקי (y - מה שבפועל נרצה שהמערכת תשערך)

הפונקציה תעזור לנו בזמן אימון הרשת לקבל אינדיקציה לגבי דיוק הרשת.

מאחר ואנחנו מאתחלים את רשת הנוירונים עם משקולות התחלתיות שהן לא אופטימליות, החיזוי הראשוני שמתקבל בתהליך הfeedforward (שכלול של הכניסות עם המשקולות) הוא לא החיזוי הטוב ביותר.

שימוש נפוץ למזעור השגיאה ברשת נוירונים הוא בעזרת Backpropagation.

המושג backpropagation כולל בתוכו משפחה של אלגוריתמים, שהרעיון המרכזי שלהם הוא למזער את השגיאה ולהביא את המשקולות למצב אופטימלי בצורה מהירה יחסית, אפילו עבור רשת עם כמות גדולה של נוירונים.

ראשית, נתאר כמה מושגי יסוד עבור האלגוריתם:

Sample

דגימה בודדת מתוך סט המידע, בדרך כלל הוא זה שיוחזר לנו מתוך המחלקה שאחראית על עיבוד סט המידע מתוך פונקציית הget item, יכול להיות למשל ווקטור באורך קבוע מתוך STFT של אות.

Batch

מספר קבוע שמייצג קבוצה של דגימות מתוך סט המידע, בדרך כלל חזקה של 2, את סט האימון נכניס לתוך הרשת בתוך Batch של דגימות, כאשר גודל הBatch נקבע באמצעות היפר-פרמטר בשם Batch Size שיקבע באצמעות סט הקרוס ולידציה.

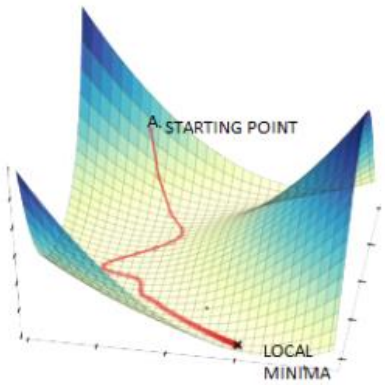
Gradient Descent and Backpropagation

זוהי שיטת אופטימיזציה איטרטיבית מסדר ראשון למציאת מינימום מקומי של פונקציה.

בשיטה זו, נעשה צעד נגדי לגרדיאנט ביחס לנקודה הנוכחית.

תהליך צמצום השגיאה פועל בצורה הבאה על כל Batch בנפרד:

- **Feedforwarding** - מזין את הtraining data ולאחר מכן מקבלים חיזוי.
- **Backpropagation** - האלגוריתם מחשב את הנגזרות החלקיות בעזרת כלל השרשרת, מפונקציית השגיאה לנוירון מסוים וכך בודק את גודל ההשפעה של המשקולות על החיזוי.
- **Weight Update** –מבצעים Backpropagation באמצעות Gradient Descent .



תיאור של מציאות המינימום באמצעות האלגוריתם

נוסחאת עדכון המשקולות:

$$\mathbf{W}_i = \mathbf{W}_i - \alpha \frac{\partial f(\{\mathbf{W}_i\}, \{\mathbf{b}_i\})}{\partial \mathbf{W}_i}$$

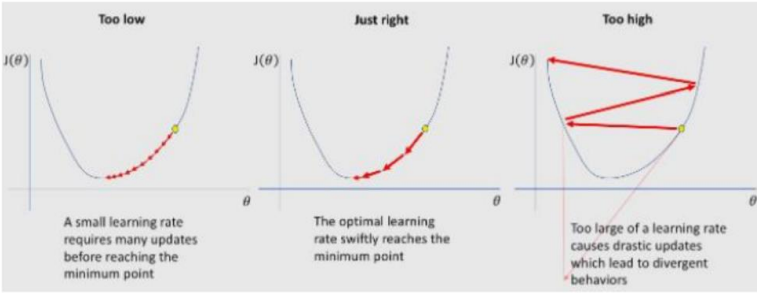
כאשר W_i מייצג משקולת והחישוב מתבצע בעזרת הגרדיאנט.
את הרשת נרצה לאמן בצורהד איטרטיבית, עבור סט רחב של דוגמאות, על פי הדרך שהצגנו.
כעת נציג מושגים שבאמצעות נממש את האימון של הרשת על גבי סט הדוגמאות:

Epoch

מתאר איטרציה על כל סט הדוגמאות שלנו, Epoch Size יהיה מספר האיטרציות שנקדיש לטובת שלב האימון של הרשת, זהו היפר-פרמטר נוסף אותו נקבע על פי הקרוס-ולידציה.

Learning Rate

קצב הלמידה הינו היפר-פרמטר גם כן, הוא קובע את קצב ההתקדמות של הגרדיאנט בכיוון המינימום, ניתן להבין שקצב גבוה מדי עלול לגרום לקפיצות מעל המינימום ולחוסר התכנסות לתוכו, וקפיצות קטנות מדי עלול לגרום לנו להתקע על מינימום מקומי ולא על מינימום גלובלי.
ישנם אלגוריתמים בשם Scheduling אשר מאפשרים שינוי קצב הלמידה כתלות בהקדמות אימון הרשת.



תרחישים אפשריים עבור פרמטרי Learning Rate שונים

Adam Optimizer

בפרוייקט זה אשתמש באופטימייזר מסוג Adam לצורך אימון הרשת, קיימים מגוון של אופטימייזרים, אך עבור פרוייקט זה נעדיף לעבור עם Adam מכיוון שהוא משלב בתוכו יתרונות של שיטות אחרות, Adam משתמש בממוצע של המומנט השני (שונות) בשונה מאופטימייזר RMSProp למשל, המתאים לעצמו את קצב למידת הפרמטרים על סמך ממוצע אמפליטודות הגראדינט שזהו המומנט הראשון (ממוצע).

האלגוריתם עובד בצורה הבאה :

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t$$
$$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2$$

א – moving average על הגראדינט והגרדיאנט בריבוע

$$\hat{m}_t = \frac{m_t}{1-\beta_1^t}$$
$$\hat{v}_t = \frac{v_t}{1-\beta_2^t}$$

ב – משערכי מהמומנט הראשון והשני לאחר נרמול

$$w_t = w_{t-1} - \eta \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}$$

ג – עדכון המשקולת

יתרון נוסף של אלגוריתם Adam הוא שגודל צעד העדכון אינו משתנה כתלות במגניטודה ולכן יכול לעזור במצבים של Saddle point .

עוד מושגי בסיס מעולם רשתות הנוירונים :

סט אימון

מטרת סט זה הינה עדכון משקלי רשת הנוירונים באמצעות Gradient descent.

סט ולידציה (קרוס-ולידציה)

סט זה מוגדר לצורך הערכה ודיוק המערכת.
ההערכה תתבצע לאחר כל איטרציה (epoch), כאשר בכל אחת מועבר מידע בגודל batch-size (מוגדר מראש ככמות מידע המועברת בכל איטרציה).

סט מבחן

סט זה יבחן את ביצועי המערכת עבור מידע חדש שלא אומנה לפיו.

Fully connected and MLP

MLP הינה רשת מסוג fully connected אשר בה כל הכניסות מחוברות לכל יחידות האקטיבציה של השכבה הבאה, רשת זו בעלת 3 שכבות- שכבת קלט, שכבה נסתרת ושכבת פלט. כמו כן משתמשת בbackpropagation לצורך אימון הרשת.

רקע תאורטי על עיבוד האות בפרוייקט

STFT – Short Time Fourier Transform

ראשית נגדיר את התמרת DTFT למעבר אות ממישור הזמן אל מישור התדר:

$$X_{DTFT}(e^{j\omega}) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\omega n}$$

התמרת DTFT מקבלת אות ומתמירה את כולו בבת-אחת. עבור התמרה זו אין התייחסות להרכב תדרי המשתנה בזמן המתאים לאותות לא סטציונרים, לכן שימוש בהתמרה זו אינו מתאים לאותות דיבור.

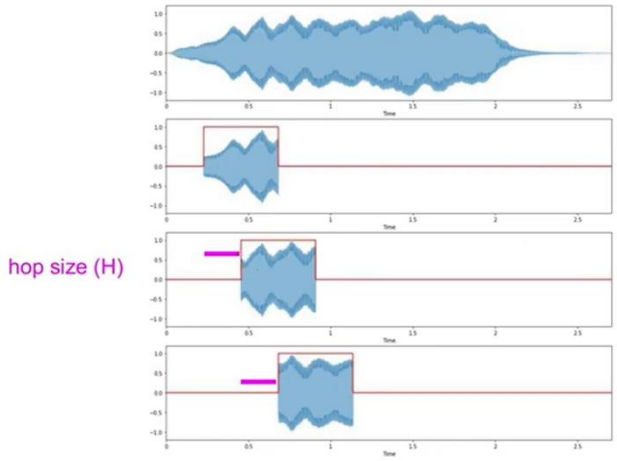
על מנת להתמיר את את הדיבור, נשתמש בהתמרה מסוג Short Time Fourier Transform (STFT)

אשר מוגדרת באופן הבא:

$$X_{STFT}(e^{j\omega},n) = \sum_{m=-\infty}^{\infty} x[m]g[n-m]e^{-j\omega n}$$

עבור $x[n]$ אות אינסופי תמך בזמן בדיד ו- $g[n]$ פונקציית חלון אנליזה בעלת אורך L.

נחלק את האות למקטעים קטנים, בכל פעם נבחר במקטע בודד ונבצע לו DTFT. לאחר מכן נעבור למקטע הבא על ידי קידום הפרמטר n, הכולל חפיפה של 50% (hop size) בין הקטעים כפי שנראה בתרשים הבא:



המחשה של דילוג וחפיפה באלגוריתם הSTFT

התמרה זו מאפשרת לנו להביט בשינוי התדרי של האות זמני עבור אינטרוול זמן מסוים, כך שלאחר מעבר על כלל החתיכות נתמיר את האות הכולל.

ככל שאורך החלון גדול יותר - נקבל רזולוציה טובה יותר של האות בתדר, היות והתמרת החלון קרובה יותר להלם (עבור חלון אינסופי אנו מקבלים למעשה את התמרת ה-DTFT הרגילה של האות). ככל שהחלון יותר קטן הרזולוציה בתדר נמוכה יותר וגדולה יותר בזמן.

לאחר סיום עיבוד האות נתמיר אותו חזרה לממד הזמן על ידי ISTFT – יפורט בהמשך.

רקע תאורטי על העשרת ספקטרום

אל המערכת נכנסים אותות דיבור הכוללים רעש. אותות אלו מורכבים מיחסים שונים של עוצמת אות הדיבור אל מול עוצמת הרעש הכולל (**SNR – Signal to Noise Ratio**), אודות סט זה הרחבנו בפרק אימון הרשת.

קיימות מספר שיטות עיקריות לשערוך ספקטוגרמות אות נקי מתוך אות רועש.

נדון בשתי שיטות עיקריות:

Masking Based
Spectrogram Enhancement

בכל השיטות בפרוייקט זה, נעבוד במישור הלוג של הספקטרום.

Masking Based

גישה זו מתבסס על שערוך מסכה שבאמצעותה ניתן לסנן את האות.
ישנן 2 שיטות לשערוך האות- מסכה קשה IBM ומסכה רכה IRM. כאמור, במוצא של שיטות אלו נקבל מסכה אותה נכפיל באות הרועש והתוצאה תהיה האות הנקי מרעשים.

בצורה מתמטית במישור:

$$X(\omega,t) = S(\omega,t) + W(f,t):$$
$$\hat{S}(\omega,t) = MASK \cdot X(\omega,t)$$

כאשר $x[n]$ הינו האות הרועש ו$\hat{s}[n]$ זהו האות לאחר המסכה.
בפרויקט נביא לידי ביטוי שתי שיטות מבוססות מסכה: IRM ו- IBM.

נזכור כי בפרוייקט זה אנחנו עובדים במישור הלוגי של הספקטרום ולכן החישוב יהיה מעט שונה עבור השערוך של האות הנקי בעזרת המסכה.
מאחר וכפל בתוך הלוג הופך לחיבור של הלוגים, נפתח את הנוסחא הבאה:

$$\hat{S}(\omega,t) = MASK \cdot X(\omega,t) + \ (1 - MASK)\ (X(\omega,t) - \beta)$$

כאשר בטא הינו היפר-פרמטר אותו נקבע על ידי סט הקרוס-ולידציה.

IBM – Ideal Binary Mask (Hard Mask)

עבור האות המותמר, שיטה זו תמדוד את יחס הSNR בנקודת זמן-תדר בספקטרוגרמה מסוימת, ותפיק ערך בינארי 0 עבור SNR נמוך או 1 עבור SNR גבוה.
כלומר, עבור נקודת זמן-תדר בספקטרוגרמה בו ערך הSNR היה גבוה, נוכל לומר כי אנרגיית אות הדיבור היא הדומיננטית במיקום זה, ערך הIBM יהיה 1 ומוצא המערכת תהיה ערך מסוים.

במצב ההפוך כאשר תוצאת IBM תיהיה אפס, כלומר שהמקטע רועש- תוצאת הכפל תתאפס ולא נשתמש בקטע זה.

נוסחה לשיטה זו :

$$IBM(t,f) = \begin{cases} 1 & \text{if } |S(t,f)|^2 > |N(t,f)|^2 \\ 0 & \text{otherwise} \end{cases}$$

יתרון שיטה זו:
לפי שיטת שערוך זו, החיתוך מתבצע בצורה גסה ולכן מתבצע סינון מוחלט של רעשים,כלומר מתקבל אות מובן יותר.

חסרון שיטה זו:
שיטה זו חותכת בצורה גסה רעשים, לכן עבור אות רועש יתכן מצב בו נסנן תדרים רלוונטים של

הדיבור ונקבל אות איכותי פחות.

IRM – Ideal Ratio Mask (Soft Mask)

שיטה זו פועלת באופן דומה לקבלת החלטה בהתאם ליחס הSNR , אך בניגוד לIBM שעבורו הערכים הם 0 או 1, עבור IRM יתקבלו ערכי ביניים המייצגים את היחסים השונים בין הדיבור לרעש, ובכך נקבל מסכה רכה.

הנוסחה לשיטה זו :

$$IRM(t,f) = \sqrt{\frac{|S(t,f)|^2}{|S(t,f)|^2 + |N(t,f)|^2}},$$

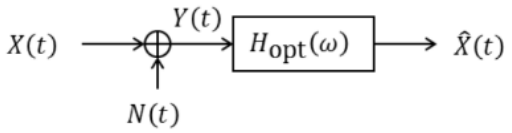
יתרון שיטה זו:

מאחר והחישוב של IRM ממפה ערכים לטווח של [0,1] המסנן יבצע סינון יותר עדין, כלומר במקום בו יש פחות רעש נקבל פחות סינון, לכן לפי שיטה זו נקבל אות דיבור יותר איכותי וגם יותר מובן.

חסרון לשיטה זו:

עבור אות חסר קורולציה ברעש, השערוך שיתקבל יהיה איכותי ומובן יותר.
שיטה זו לא תוריד מספיק את הרעש משום שתוצאת ההכפלה אינה מתאפסת.

שיטה זו דומה למסנן וינר, אשר מוגדר באופן הבא:
עבור כניסת אות $X(t)$ ורעש $N(t)$ מתקיים :



$$H_{opt}(\omega) = \frac{s_{xx}(\omega)}{s_{xx}(\omega) + s_{NN}(\omega)}$$

Spectrogram Enhancement (Mapping Based)

בשיטה זו נכניס אות רועש בתחום STFT ונשערך את האות הנקי ללא שימוש במסכה או שיטות שונות.
במוצא השיטה יתקבל חיזוי ולפי השוואה למוצא האות המקורי, נקבל שגיאה שבעזרתה נוכל לעדכן את המשקולות לפי הGradient Descent .

יתרון שיטה זו :

חסינות יותר גבוהה בפני רעש.

חסרון שיטה זו :

צריך הרבה פרמטרים. החיזוי שמתקבל יהיה בעל distortion גבוה יותר משל השיטות האחרות.

המערכת בעלת שני שלבים: שלב אימון ושלב ההעשרה.

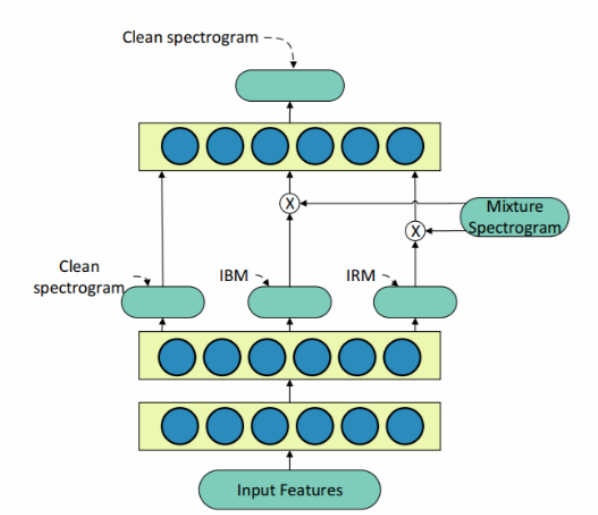
הפתרון הממומש במאמר

המאמר מציע כפתרון לבעיה רשת עמוקה, שמטרתה לקחת כמה שיותר יתרונות וכמה שפחות חסרונות מכל השיטות שהוצגו ולשקלל אותן לטובת שערוך אופטימלי של האות הנקי.
בכל פעם מכניסים המערכת קולטת פריים של דגימות לצורך העיבוד, כאשר על כל פריים אנו נבצע כל אחת משלושת השיטות הבאות:

1. IBM – Ideal Binary Mask
2. IRM – Ideal Ratio Mask
3. שערוך ישיר של הספקטרוגרמה הנקייה.

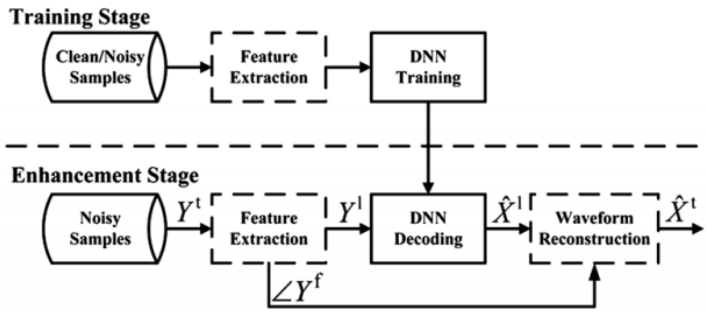
בנוסף, נשרשר את כל אחד מהשערוך המתקבלים עבור כל שיטה, ונעביר דרך שכבה נוספת ברשת, ונקבל שערוך עבור שלושת השיטות ביחד.

שלושת השיטות הינן שיטות שונות שמטרתן המשותפת הינה לשערך את האות דיבור הנקי.
שיטות הIRM והIBM מייצרות מסכה רכה וקשה, בהתאמה, עבור הסקטרוגרמה, כך שהאות הנקי יתקבל כתוצאה מהכפלה של האות הרועש במסכה.
לכל שיטה יתרון שערוך משלה ויכולת להתאים למקרה אחר של בעיית העשרת דיבור ולכן, המערכת היא יעילה מכיוון שהיא יודעת לבצע שקלול חכם ולתת משקל רב יותר למוצא עם השגיאה הקטנה יותר ומשקל נמוך יותר למוצא בעל השגיאה הגדולה יותר.



סכימה של הרשת, ניתן לראות כי סה"כ ישנם ארבעה מוצאים לרשת.
מוצא עבור כל אחת מהשיטות ובנוסף את המוצא המשוקלל.

לאחר שקלול שלושת השיטות, נשתמש באלגוריתם הISTFT על מנת להרכיב בחזרה את האות הנקי.
מאחר שאנחנו עובדים עם הערך המוחלט של ההתמרת פורייה, נרצה לשמור את הפאזה של ההתמרה עבור כל פריים, ובשלב השחזור של האות נצמיד כל פאזה אל הפריים אלייה היא שייכת ובכך נוכל לבצע את השחזור בצורה מלאה.
נרצה לציין כי בפרוייקט זה ההתרכזות היא עבור שערוך המגניטודה, נשים לב כי אנו לא משערכים את פאזת האות הנקי וכי אנחנו משתמשים בפאזה הנקייה.
למרות זאת, בדקתי מה קורה כאשר משתמשים בפאזה מהאות הרועש, והתוצאות היו דומות לאוזן וכמעט לא היה ניתן להבחין בהבדלים, לכן בקירוב השערוך הוא שערוך אבסולוטי של האות הנקי.



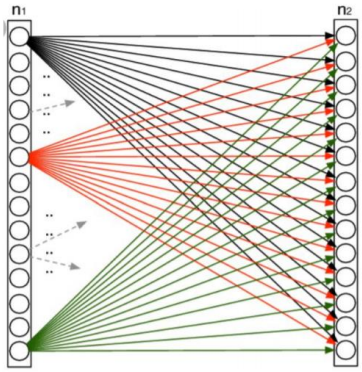
תיאור הליך הסינטזה של האות המשוערך, כולל הצמדת הפאזה.

רקע על שכבות שונות ברשתות נוירונים עמוקות

אתמקד בפירוט על השכבות השונות שמימשתי ברשת של פרוייקט זה.

Fully Connected Layer

שכבה בה כל מוצא מהשכבה הקודמת מחובר לכל כניסה של נוירון בשכבה.
במודלים שונים, השימוש בשכבה זו הוא בעיקר עבור השכבה האחרונה, המשלבת את הנתונים שחולצו מהשכבות שלפנייה על מנת ליצור את השערוך הסופי.
החיסרון בשיטה זו הוא שחיבור כל כניסה למוצא הוא בזבזני ועלול לעלות בחישובים רבים.



תיאור הליך הסינטזה של האות המשוערך, כולל

Dropout layer

שכבה זו מאפסת נוירונים באופן אקראי בשלב האימון בלבד, כך בעצם נמנע מהם לעדכן את המשקולות. על ידי קביעת הפרמטר p נקבע אחוז מסויים של נוירונים שלא יופעל בכל שכבה, כך שנקבל רשת אפקטיבית קטנה יותר.
המטרה של שכבה זו היא למנוע את המצב של overfitting, איפוס של נוירון אחד גורם לנוירון אחר לבחור במקומו וכך הרשת נהיית פחות רגישה למשקולות ספצייפים ויותר גנרית עבור המידע.

אימון הרשת

השלב הראשון שארצה להגדיר עבור אימון הרשת הוא שלב אתחול הhyper-parameters עליהם הרחבתי בפרקים הקודמים בספר.
בשלב זה אגדיר פרמטרים התחלתיים עבור המערכת (בהתחלה לפי המאמר ואחכ לפי ניסוי וטעייה).

Batch Size – 256
Hidden Layer Size – 1028
p of Dropout Layer – 0.2
Optimizer – Adam
Learning Rate – 0.001
Epoch Size – 200

פונקציית הפסד (Loss Function)

בפרוייקט זה נבחר להשתמש בפונקציית Loss מסוג MSE .
המטרה של הפונקציה היא לחשב את השגיאה בין המוצא המשוערך לבין האות הנקי ולכמת לנו אותו למספר, כאשר נרצה להקטין את המספר הזה שישאף ל-0.
האינטרס להשתמש דווקא בפונקציה זו נובע מהגדרת הבעיה, מאחר ואצלנו אנחנו לא מבצעים סיווג אלה רגרסיה, נרצה פונקציית שגיאה מתאימה, ולכן MSE תתאים לנו במקרה הזה.

הגדרת הפונקציה :

$$\mathrm{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

ניהול ה dataset

החלוקה של הdataset מתבצעת כך:

Training set – סט האימון

3000 סאמפלים שונים של אותות דיבור נקיים מתוך מאגר TIMIT
קטע רועש באורך של כ-3 דקות המכיל סוגים שונים של רעשים (צרי סרט, רחבי סרט וכו..)

Validation set – סט הולידציה

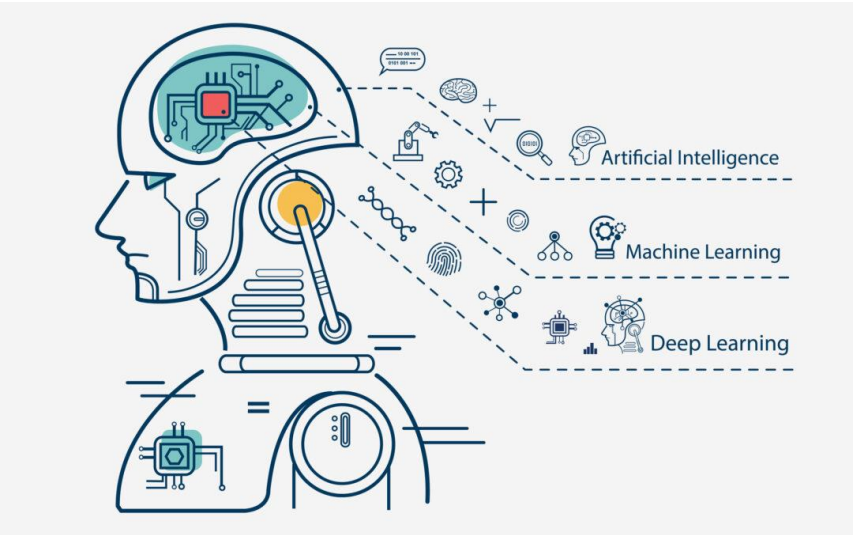
חיתוך של 20% מתוך כלל סט האימון לטובת סט ולידציה.

Test set – סט המבחן

עבור הסט הזה, נבחר הרבה דוגמאות של אותות דיבור נקיים שונים מתוך המאגר וקטע ארוך של רעש שונה מהקטע שהשתמשתי בו בשלב האימון.
בשלב הטסט, אשתמש גם ברעשים צרי סרט ורחבי סרט על מנת למדוד את ביצועי המערכת עבור מקרים אלו בצורה פרטנית.

הגדרת הבעיה

כיום, הצורך ביכולת של הבנת שמע בעזרת אמצעים דיגיטליים, גובר, מאחר ורוב האמצעים היום הם דיגיטליים במימושם.
עם צמיחת הענף של למידת מכונה כך גם נבנה הצורך במערכות הנשענות על רשתות אלו על מנת לבצע פעולות כמו שערוך אות שנעשה בפרוייקט זה, וכמו בעיות רבות אחרות.
אם עד היום לא היה את המענה הטכנולוגי בהבנה של הנאמר מתוך אות רועש כלשהו, תחום הלמידת מכונה עונה על בעיה זו ודיי בקלות, ועוזר לסנן רעש מתוך אות ולשערך אות נקי.
היתרון הוא עצום, הפשטות היחסית שבמימוש, הביצועים שניתן להגיע אליהם והפוטנציאל העסקי של המימוש גורמים לתחום של למידת מכונה להוות תחליף בלעדי בהרבה תחומים למה שעד היום הסתמך על החלטה אנושית.
בנוסף ניתן לראות כי העולם מתקדם בכיוון ה'ענן', רוב התוכן הוא נגיש לכולנו ולכן קל לגבש דוגמאות לצורכי אימון רשתות, כמו לדוגמא גוגל, אשר מאמנת את הרשתות שלה לפי אינפוטים של משתמשים, כאשר אנו מעלים תמונה לדרייב.
באותה מידה אנחנו תמיד יכולים להרחיב את סט האימון שלנו, תמיד יכולים למצוא דוגמאות יותר מגוונות ובכך לבנות לעצמנו תשתית עוצמית יותר ללמידת מכונה.
בפרוייקט זה, מוצג מודל סינון רעשים מתוך אות רועש כלשהו אשר ממומש על גבי רשת נוירונים מאומנת, הפרוייקט מקבל כקלט אות רועש ומשערך את האות הנקי מתוך האות הרועש.
השימוש בכלי כזה יכול לבוא לידי ביטוי במגוון תחומים: משטרתי, עולם הסאונד, טלפונים וכו...
המטרה של הפרוייקט היא לדעת לבצע הפרדה בין אות דיבור לרעש באמצעות אימון של רשת, את ביצועי הרשת אמדוד באמצעות סטנדרטים מסויימים (STOI, PESQ) אך חשוב לזכור כי אם ארצה, תמיד אוכל לשפר את ביצועי הרשת, תמיד אוכל למצוא עוד דוגמאות מגוונות ולהתאמן עליהן ותמיד אוכל לשפר את החומרה שמבצעת את העיבוד ולזרז או לתמוך באימונים יותר כבדים של הרשת.



עיבוד המידע

את המידע נרצה לעבד בצורה שנוכל להעבירו דרך הרשת ולקבל את השגיאה ואת השערוך הרצויים. הדרך שבה אעבד את המידע היא בצורה של מסגרות (פריימים). תחילה, אנו נרצה להכניס לרשת אות שהוא רועש, אך במקביל נרצה לדעת מיהו האות הנקי של אותו האות הרועש, ולכן נרצה לשמור את כל המידע הזה.

נבחר בצורה אקראית אות דיבור נקי כלשהו, בנוסף נגדיר אות רועש על ידי בחירה של קטע רועש אקראי באורך של האות הדיבור הנקי, מתוך המאגר הרועש הארוך, ונצמיד אותו לאות הנקי.

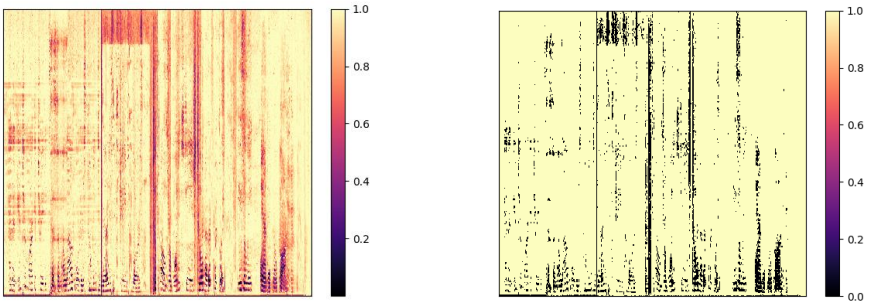
בנוסף, נבחר ערך SNR אקראי מתוך רשימת ערכי SNR – במקרה שלנו [5, 0, -5] ונעבד את האות כך שערך הSNR שבחרנו יתאים לאות הרועש.

כעת נבצע התמרת STFT לאותות (כולל לוג) ובנוסף נדאג לנרמל את האות הרועש בסיום התהליך. ע"י בחירת חלון Hanning בגודל של 512 וחפיפה בגודל של 75 אחוז, נקבל התמרה שגודל כל מסגרת STFT שלה הוא 257.

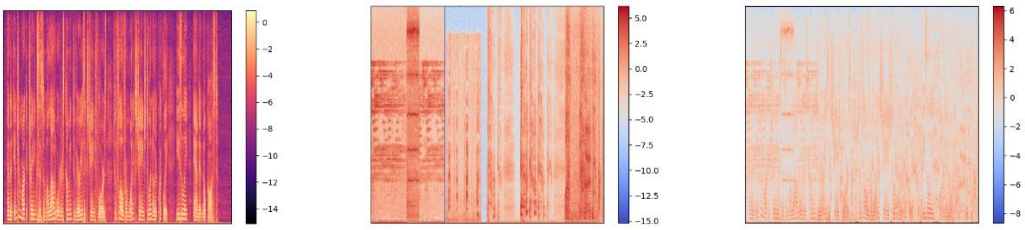
על מנת למטב את ביצועיי הרשת, נרצה לספק לרשת מידע נוסף על האות הרועש, נעשה זאת בצורת Context Frames. הרעיון הוא לשרשר 2 מסגרות מכל צד של מסגרת בודדת של האות הרועש, כלומר: ניקח מסגרת מתוך האות הרועש, נוסיף לו את השתי מסגרות הקודמות לו, ואת השתי מסגרות שאחריו ונקבל שגודל הContext Frame הוא:

$$2 * 257_{frames\ before} + 257_{current\ frame} + 2 * 257_{frames\ after} = 1285$$

כעת, נבצע את החישוב של המסכות IBM ו-IRM באמצעות הרעש ואות הדיבור הנקי על ידי נוסחאות מסויימות כפי שהוצג בפרקים הקודמים.



תוצאות של חישוב המסכה אותה נרצה לשערך עבור אות נקי ואות רעש, מימין זו מסכת IBM ומשמאל IRM.



האות הנקי (משמאל), אות הרעש (אמצע) והאות הרועש (ימין)

מימוש סינון ועיבוד של סט המידע

הפרוייקט ממומש באמצעות Python ובשילוב עם קודי MATLAB, כאשר השימוש העיקרי לצורך עבודה עם רשת נוירונים הוא בעזרת ספריית PyTorch הפופולרית,
לצורך השלב הראשוני, נדרש מימוש של פונקציות עיבוד אות על מנת לחתוך קטעי דיבור ורעש, לבצע התמרות ולהטמיע SNR .
אגדיר את הפונקציות הבאות לצורך עיבוד אות:

```
# get_list_of_files
# ==========================================================================================
def get_list_of_files(path)


# get_sample_from_file
# ==========================================================================================
def get_audio_from_path(audio_path)

# get_noise_segment
# ==========================================================================================
def get_noise_segment(noise_audio, clean_length)


# multiply_noise_by_SNR
# ==========================================================================================
def multiply_noise_by_SNR(clean_signal, noise_segment, desire_snr)


# get_list_of_wav_files
# ==========================================================================================
def get_list_of_wav_files(path_to_files)


# y_stft
# ==========================================================================================
def y_stft(z, K, overlap)


# istft
# ==========================================================================================
def istft(A, P, synt_win)
```

לאחר שלב זה, אגדיר את המחלקה תחלק את dataset לתוך פריימים ובנוסף תבצע חישובים של המסכות, על מנת ליעל את צורת בניית הdataset וכתוצאה מכך את זמן האימון, ערכתי השוואות בין סוגי פעולות שונות על list בפייתון, ובסופו של דבר השתמשתי בפעולה של extend על ליסט שאיטרציה עבורה עולה לי כ2ns

דוגמא: מימוש הContext Frame

```
def make_context(noisy, context_mat):
    noisy_pad = np.zeros((frame_size, 2))
    noisy = np.append(noisy, noisy_pad, axis=1)
    noisy = np.insert(noisy, 0, 0, axis=1)
    noisy = np.insert(noisy, 1, 0, axis=1)
    iter_range = (len(np.transpose(noisy)))
    alist = []
    for t in range(2, iter_range - 2):
        if t % 1000 == 0:
            print("{}/{} Context frames loaded.".format(t, iter_range - 4))
        X = noisy[:, [t - 2, t - 1, t, t + 1, t + 2]]
        X = X.flatten()
        alist.extend(X)
    context_mat = np.reshape(alist, (iter_range - 4, 5 * frame_size))
    return np.transpose(context_mat)
```

בניית הרשת

הרשת מורכבת משלושה שכבות נסתרות, שכבת כניסה ושכבת מוצא.
לרשת ישנם ארבעה מוצאים שונים אותם נשווה בהמשך על מנת להבין את היתרון והחיסרון של כל שיטה.

```
class NeuralNetwork(nn.Module):
    def __init__(self, input_size=5*257, hidden_size=1024, stft_size=257):
        super(NeuralNetwork, self).__init__()
        # hidden 1:
        self.fc1 = nn.Linear(input_size, hidden_size)

        # hidden 2:
        self.fc2 = nn.Linear(hidden_size, hidden_size)

        # hidden 3:
        self.fc3 = nn.Linear(hidden_size, stft_size)

        self.dropout = nn.Dropout(0.2)

        self.sgm = nn.Sigmoid()

        self.mlp = nn.Sequential(
            nn.Linear(stft_size * 3, 1600),
            nn.ReLU(),
            nn.Dropout(0.2),
            nn.Linear(1600, 1600),
            nn.Linear(1600, stft_size),
        )

    def forward(self, x):
        x = F.relu(self.fc1(x))
        x = self.dropout(x)

        x = (self.fc2(x))

        spec = (x)

        ibm = self.sgm(x)
        irm = self.sgm(x)

        ibm_input = torch.mul(spec, ibm) + torch.mul((1 - ibm), (spec - 1.4))
        irm_input = torch.mul(spec, irm) + torch.mul((1 - irm), (spec - 1.4))



        total = torch.cat((spec, ibm_input, irm_input), 1)

        total = (self.mlp((total)))

        return total, spec, ibm, irm
```

הכניסה לרשת מוגדרת להיות בגודל של input_size=1285 שזה הגודל של Context Frame איתו אנחנו עובדים, בנוסף גודל המוצא הוא 257 שזה גודל של פריים רגיל של אות נקי משוערך
ניתן לראות כי כל השכבות מחוברות בצורה של Fully Connected , כאשר פונקציית האקטיבציה מהשכבה הראשונה לשנייה מוגדרת להיות ReLU בגודל של 1024 nodes, בשכבה השנייה מתבצע פיצול לשיטות השונות IBM, IRM, Clean Spectrogram, לכן, עבור שיטות שהן מסוג מסכה, הפעלתי את הסיגמוייד בתור פונקציית האקטיבציה, ולשערוך מצורת מיפוי, הפעלתי פונקציית ReLU.
לאחר מכן, הרשת מבצעת חישוב של השערוך בעזרת המסכות כפי שניתן לראות.
בנקודה זו קיימים שלושה שיערוכים שונים לרשת, נבצע שרשור של שלושת השערוכים בעזרת torch.cat ונקבל מטריצה שגודל כל פריים שלה הוא $257*3 = 771$.
את המטריצה המשורשרת נכניס לתוך השכבה השלישית שהיא שכבת MLP שמורכבת מ Linear, ReLU and Dropout.
ניתן לראות כי הרשת מחזירה ארבעה מוצאים שונים:

Total – השערוך של השרשור.

Ibm – השערוך לפי IBM.

Irm – השערוך לפי IRM.

Spec – השערוך לפי חישוב ישיר של האות הנקי.

נחבר את ארבעת המוצאים, נחשב את הגרדיאנט, ונבצע את אלגוריתם הBackpropagation.

```
# Compute prediction and loss
pred_total, pred_spec, pred_ibm, pred_irm = model(spec)

output_loss = loss_fn(pred_total, y)
loss_spec = loss_fn(pred_spec, y)
loss_ibm = loss_fn(pred_ibm, ibm_mask)
loss_irm = loss_fn(pred_irm, irm_mask)

total_loss = loss_spec + loss_ibm + loss_irm + output_loss

# Backpropagation
optimizer.zero_grad()
total_loss.backward()
optimizer.step()
```

תוצאות

הרשת בפרוייקט, מתאמנת 200 איטרציות על הdata set.
בסיום האימון הLoss והValidation מגיעים לאיזור רווייה שנמדד בגודל של כ3.5, זאת לאחר אופטימיזציה של הרשת בעזרת הCross Validation, נזכיר כי הLoss מוגדר מחיבור של שלוש שיטות שיטות להעשרת דיבור ולכן המספר עצמו מורכב מארבעה לוסים שונים שכל אחד נאמד בערך על 0.9.
לצורך ההשוואות יצרתי רשתות יעודיות על מנת לספק את המוצא של הרשת של הפרוייקט ובנוסף את המוצא עבור IBM, IRM, Clean Spectrogram.
ניסיתי לקבל תוצאות שונות גם עבור אימון של הSpectrogram ללא שימוש בLog, והרשת אכן הגיעה למספרי Loss נמוכים יותר אך הביצועים בפועל היו פחות טובים, לכן החלטתי פחות לנסות להוריד את הLoss ולהתעסק יותר בלקבל ביצועים טובים.
את ביצועי העשרת הדיבור נעמוד במונחים סטנדרטיים של STOI ו- PESQ,

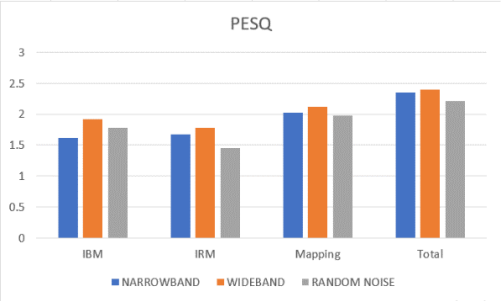
PESQ – Perceptual Evaluation of Speech Quality

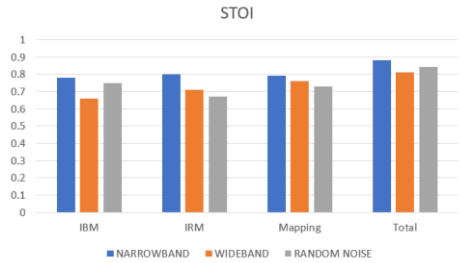
משפחה של תנאים סטנדרטיים אשר נמדדים בעזרת טסטים אוטומטיים ואומדים את איכות שערוך האות הנקי, התוצאות נאמדות בטווח מספרים בין 1 ל-5 כאשר מספר גבוהה יותר אומר שערוך טוב יותר במובן של איכות השמע.
האלגוריתם מקבל את אות הReference – האות הנקי, את האות המשוערך, והאם מדובר במקרה של Wideband או Narrowband .

STOI – Short Time Objective Intelligibility

בשונה מPESQ, מבחן זה אומד את איכות הבנת השמע מתוך השערוך, הערכים המתקבלים הם בטווח של 0 עד 1 כאשר מספר גבוהה יותר משמע שיערוך מובן יותר.

בפרוייקט זה נמדדו התוצאות עבור כל אחת מהשיטות לשיערוך, על מנת לדייק את התוצאות כמה שניתן, בדקתי את הטסטים הנ"ל על כמה שיערוכים שונים ולאחר מכן עשיתי ממוצע.
התוצאות שהתקבלו הן עבור אותות מורעשים כאשר הSNR הוא משתנה אקראי שמקבל את הערכים 5 ,0 ,5- בהסתברות שווה.

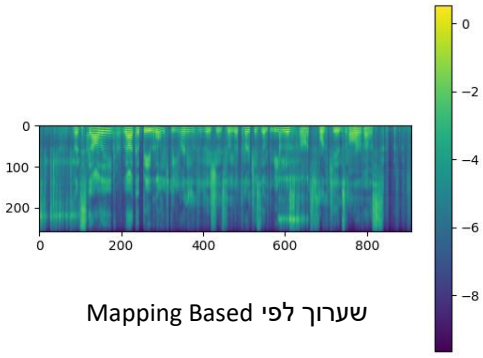




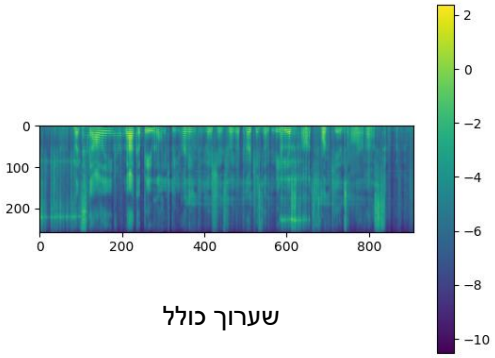
תוצאות של PESQ ו- STOI

ניתן לראות כי תמיד המוצא הכולל נותן את הביצועים הטובים ביותר, זאת מכיוון שהרשת יודעת לתת משקל עבור השיטה האופטימלית עבור כל מקרה, ולכן מבצעת אופטימיזציה לשערוך מכאן שקיימת כדאיות בשרשור של שלושת השיטות לצורך שיערוך של אות רועש.
על מנת להמחיש זאת בצורה טובה יותר, אראה את תמונות הספקרוגרמה המתקבלות בכל שערוך ובנוסף של האות הנקי.

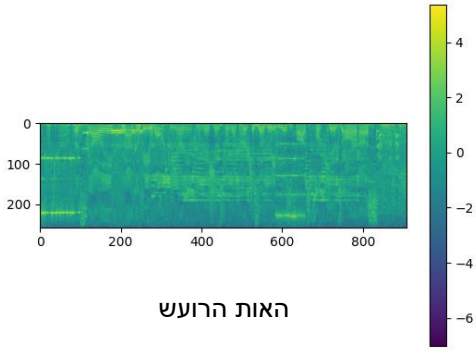
דוגמא של שערוך עבור רעש מסוג Wideband :



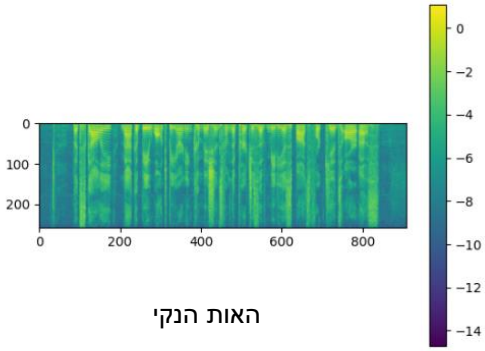
שערוך לפי Mapping Based



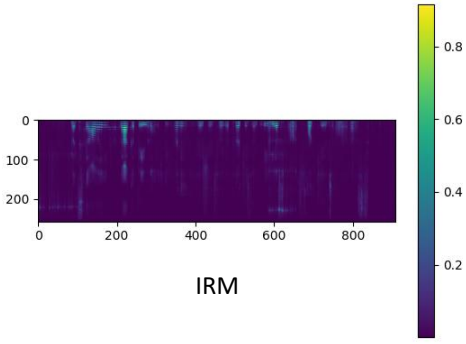
שערוך כולל



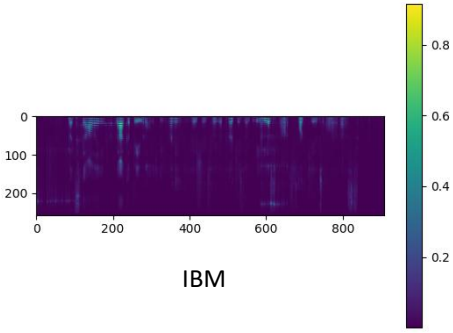
האות הרועש



האות הנקי

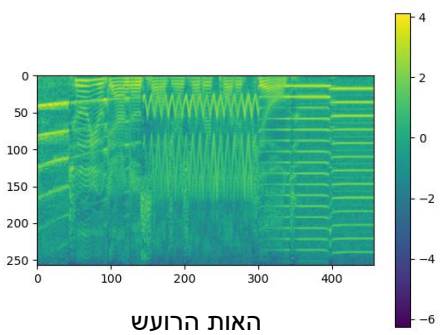


IRM

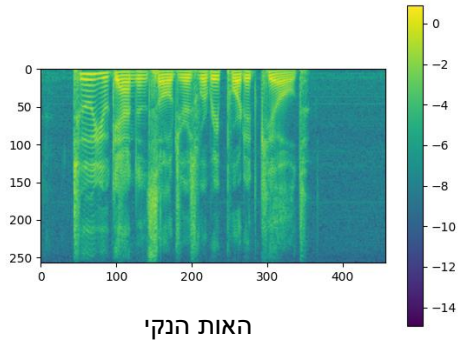


IBM

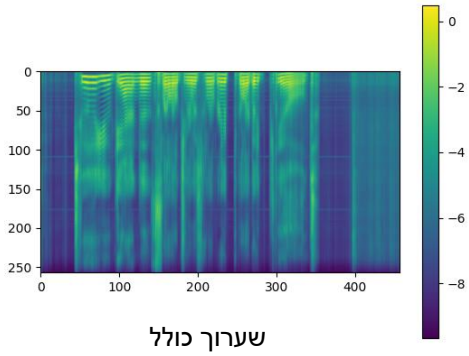
דוגמא של שערוך עבור רעש מסוג Narrowband :



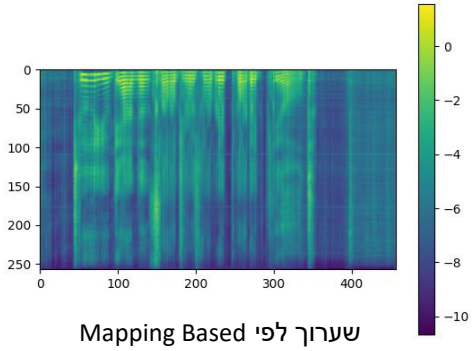
האות הרועש



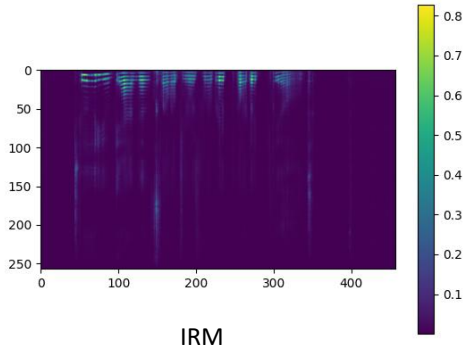
האות הנקי



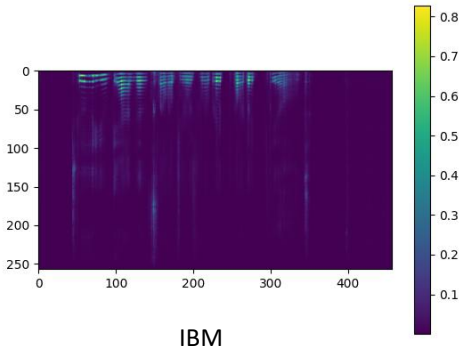
שערוך כולל



שערוך לפי Mapping Based



IRM



IBM

מסקנות ודוגמאות

ניתן לראות מהתוצאות שקיבלנו כי הרשת אכן מבצעת סינון של הרעשים, הרשת יודעת להתמודד עם מצבים של רעשי Wideband & Narrowband ועם ערכי SNR שונים.
משמיעה לאוזן, התוצאות אכן נשמעות טובות ברוב המקרים, קיימים מקרים בהם הרשת קצת מתקשה לנקות רעשים, למשל מתי שהSNR גבוה והרעש מתגבר על הקול, קשה לרשת להבחין באות של הקול הנקי מתוך הרועש.
מסקנה חשובה נוספת היא שתמיד ניתן לספק לרשת עוד מקרים ודוגמאות ולאמן אותה בצורה יותר יסודית וכך להביא לביצועים טובים יותר.
דוגמאות של ביצועי הרשת:

דוגמא ראשונה

clean.wav noisy.wav total_net_prediction.wav spec_prediction.wav irm_prediction.wav

ibm_prediction.wav

דוגמא שנייה

total_net_prediction.wav spec_prediction.wav noisy.wav irm_prediction.wav ibm_prediction.wav

clean.wav

דוגמא שלישית

total_net_prediction.wav spec_prediction.wav noisy.wav irm_prediction.wav ibm_prediction.wav

clean.wav

דוגמא רביעית

total_net_prediction.wav spec_prediction.wav noisy.wav irm_prediction.wav ibm_prediction.wav

clean.wav