Department of Computer Science

Final Project

SUBJECT NEURAL TAGGER

Finding the subject in a sentence by semantic analysis through deep learning

Students:

Michael Cohen 206798647

Tal Ades 208495572

Moderator:

Dror Mughaz

michael cohen

[Email address]

תודות

ראשית אנו רוצים להודות **לבורא עולם** שהביא אותנו לסיים בהצלחה את הפרויקט וליווה אותנו לאורך כל התואר.

תודה רבה **למר דרור מוגהץ** היקר, על העזרה הסבלנות התמיכה הליווי השיחות הארוכות והעצות הנהדרות והכל במסירות ובסבר פנים יפות. יה״ר שנזכה לעבוד עוד יחד ולהחכים במעלות התורה והמדע. בזכות העולם המופלא שדרור פתח בפנינו עוד בתחילת השנה השלישית שלנו לתואר, זכינו ואני זוכים כל יום להיחשף לעולם הData Science ולמידת המכונה. שתזכה להרבות תלמידים ושיפתחו בפניך שערי בינה חכמה ודעת.

לחנה קליין ולכל צוות מחלקת הנדסת תוכנה ומדעי המחשב, בראשם **לפרופסור בוחניק**, **לד״ר מוטי רייף ולמר אבי טרייסטמן** שליוו את כל שלבי הפרויקט באופן יוצא מן הכלל ובאדיבות רבה. אציין כי כבר בראשית הדרך כשעלה במוחנו לפתוח את ׳מועדון הסטודנטים ללמידת מכונה׳, ד״ר מוטי ופרופסור בוחניק סייעו לנו רבות להוציא זאת לפועל וע״כ ישר כוח גדול.

תודה גדולה **לפרופסור אבי רוזנפלד** על התמיכה ועל נידוב מעבדת המחשבים שברשותו.

תודה רבה לדוקטורנט ולבלשן **דוד נודל** מהאוניברסיטה העברית בירושלים, על העצות וההכוונות בתחום התחבירי והבלשנות.

תקציר

בעבר, הגבול בין משימות שמתבצעות בידי אדם ובידי כלים מלאכותיים היה ברור. עם התפתחות הטכנולוגיה ההולכת וגואה גבולות אלה החלו להיטשטש. תחום זה של משימות שעד כה היו מנוכסות לבני האדם וכרגע, הופכות למשימות שיכולות להתבצע בידי מכונה נקרא - "בינה מלאכותית" או באנגלית "Artificial intelligence".

בין התחומים הנחקרים באקדמיה ובתעשייה נמצא תחום ניתוח השפות הטבעיות. המחקרים בתחום מנסים לפתח מערכות חכמות שתוכלנה להבין טקסטים טבעיים של שפות מדוברות כמו אנגלית, צרפתית וכדו' להסיק מסקנות משמעותיות ולהפיק מהם תוצרים בעלי ערך.

בפרויקט זה התמקדנו בניתוח של משפטים בעברית, כאשר 'מטרת העל' היא לחקור ולפתח מודל ייחודי המסוגל לזהות את ה-נושא (subject) במשפטים. הרעיון של זיהוי נושא הוא מורכב מחד גיסא ובעל חשיבות גדולה מאידך. בין שלל התועלות החשובות שניתן להפיק מן מערכת שכזו הוא היכולת לייעל מערכות של סיווג, תקצור ותרגום טקסט.

בכדי להתמודד עם משימה זו השתמשנו בטכנולוגיות ושיטות מחקר מתחום ה-"Deep Learning", לאחר מחקר בנינו מודלים משלוש סוגים של רשתות נוירונים מורכבות, LSTM,GRU Bi-LSTM, כאשר ארכיטקטורת המודל היא Seq2Seq עם שילוב של Attention Mechanism. עם שילוב של Word Embedding בשיטת Word2Vec.

תוצאות המחקר הציגו אחוזים גבוהים של הצלחה. מבחינת התאמת המודל ופונקציית הLoss מכניקת ה– attention על Bi-LSTM הציגה תוצאות של 97.65 אחוז התאמה, בהרצה על-LSTM הציגה 0.82 וב-GRU 0.98 אחוז. תוצאות אלו הציגו עליונות משמעותית מבחינת מודלים קלאסיים הקיימים בתחום.

Summary

In the past, the boundary between human and artificial tools was clear. As technology evolved, these boundaries have begun to blur. Such tasks, which have hitherto been man-made and currently can be performed by a machine belong to the field of "artificial intelligence".

Among the fields explored in academia and industry is that of natural language analysis. Research in this field is trying to develop smart systems that can understand natural texts of spoken languages -- English, French and so on -- to draw meaningful conclusions and produce valuable products.

In this project, we focused on the analysis of Hebrew sentences, with the overall purpose of researching and developing a unique model capable of identifying the subject in sentences. Identifying a subject is quite complex, on the one hand, and of great importance, on the other. Among the many important benefits that can be gained from such a system is the ability to optimize systems of categorization, summarization and translation.

To tackle this task, we used deep learning technologies and research methods. After researching, we built models from three types of complex neuronal networks -- Bi-LSTM, LSTM and GRU -- with the model architecture being Seq2Seq combined with Attention Mechanism. For Word Embedding we used Word2Vec.

The results of the study achieved a high percentage of success. In terms of model fit and Loss function, Bi-LSTM with Attention showed 97.65 percent matching results, the LSTM model with Attention presented 0.82 and the GRU with Attention 0.98 percent. These results were significantly superior to classical models that exist today in the field.

תוכן העניינים

רשימת איורים וטבלאות

1 מבוא

1.1 סוגי למידה שונים

נהוג לחלק את אלגוריתמי למידת המכונה למספר סוגים:

- **למידה מונחית** (supervised learning)- כל דוגמה מגיעה יחד עם תווית סיווג. מטרת האלגוריתם היא לחזות את הסיווג של דוגמאות חדשות שאותן הוא לא פגש בתהליך הלמידה. אימון של רשת עצבית מלאכותית ("רשת נוירונים") מסתמך על אלגוריתמים מסוג זה.
- **למידה בלתי מונחית** (unsupervised learning)- מטרת האלגוריתמים היא למצוא ייצוג פשוט או תבנית להבנה של אוסף הנתונים. שיטות נפוצות מסוג זה הן חלוקה לצברים (clustering), והטלה ליריעות ממד נמוך כגון ניתוח גורמים ראשיים (PCA).
- **למידת חיזוק** (reinforcement learning)- אלגוריתם הלמידה מקבל משוב חלקי על ביצועיו (רק לאחר סיום ביצוע המטלה) ועליו להסיק אילו מהחלטותיו הם שהביאו להצלחה/כישלון.

במחקר זה נתמקד בסוג ה**למידה המונחית**. אחד המודלים המתמטיים הנפוצים בתחום זה הוא "**רשת נוירונים**". הרשת מורכבת ממספר "נוירונים" המסודרים בשכבות, כאשר כל נוירון יכול לתקשר עם מספר נוירונים אחרים במערכת. כל נוירון מסוגל לבצע פעולות חישוביות פשוטות, ובתורו להעביר את המידע שהסיק לשאר הנוירונים. באופן זה, עם ההתקדמות של המידע בשכבות, המערכת הופכת את המידע הגולמי למידע בעל ערך. כפועל יוצא מכך, המערכת לומדת לקבל החלטות מדויקות יותר. הרשת מקבלת את המידע המעובד, מבצעת הרצות של למידה ועיבוד הרשת עד לקבלת רשת שמסוגלת להוציא את הפלטים המתאימים. הבעיות שנתעסק בהן בפרויקט זה יתמקדו כאמור בניתוח של השפה העברית תוך התמקדות במשפטים מתוך השפה כאשר ברצוננו להצליח לאבחן ולזהות בכל משפט נתון את ה**נושא** (subject) עליו המשפט מדבר. המוטיבציות להשגת יכולת זו הינן רבות להלן מספר משימות שיוכלו להתייעל עם שימוש בניתוח נושא במשפטים:

1. **סיווג טקסטים** - במשימה זו האלגוריתם מקבל טקסט בנושא מסוים/ מאמר על דמות וכדו'. כתוצאה מכך שהאלגוריתם מצליח לחזות את הנושא הכללי (topic) או את הדמות עליה מדבר המאמר. למשימה זו יש חשיבות בכדי לייעל משימות גדולות יותר כמו חיפוש מאמרים, סיווג מאמרים וקישור ביניהם. המשימה שבה אנחנו נתמקד (מציאת נושא במשפט) יכולה להועיל באופן משמעותי למשימה זו של סיווג. לדוגמא אם יש לי כבר משפטים מתויגים לפי נושא (subject) האלגוריתם יוכל לזהות ביתר קלות את הדמות הראשית עליה מדבר המאמר.
2. **סיכום מאמרים** - במשימה זו האלגוריתם מקבל מאמר או טקסט מסוים ומצליח להפיק מתוך הטקסט סיכום איכותי של כל המאמר. המשימה בה נתמקד (מציאת subject) יכולה לייעל אלגוריתמים מהסוג הזה, מכיוון שההנחה היא שבסיכום ארצה להתמקד כמה שיותר בנושא (subject) כדי להצליח להעמיד סיכום איכותני שלא מפספס מילים חשובות.

1.2 הגדרות ומושגים כללים

כיצד אפשר לגרום למחשב ללמוד לבד? האם מחשב מסוגל לחקות משהו מתוך מה שאנחנו מחשיבים כ"מחשבה אנושית"? כמו למשל - היכולת להסיק מסקנות, להעלות רעיונות לקשר אירועים וכדו'.

בינה מלאכותית (נקראת גם **אינטליגנציה מלאכותית**) היא ענף במדעי המחשב העוסק ביכולתנו לגרום למחשבים לפעול בצורה המציגה יכולות שאפיינו עד כה את הבינה האנושית בלבד. הגדרה דומה לתחום זה ניתנה על ידי **מרווין מינסקי:** המדע שמתעסק ב*לגרום למכונה להתנהג בדרך שהייתה נחשבת לאינטליגנטית לו אדם התנהג כך"*. העוסקים בבינה מלאכותית מבכרים לעיתים הגדרה מצמצמת של מושג זה, כאילו בינה מלאכותית היא *"כל מה שאדם יודע לעשות אך עדיין אין אנו יודעים לתכנת אותו"*.

נעשו ניסיונות רבים לבחון מחשבים כבעלי "אינטליגנציה מלאכותית" המבחן הידוע ביניהם הוא "**מבחן טיורינג**". מבחן טיורינג - הוא כינוי למבחן שהציע אלן טיורינג בשנת 1950, במאמר שפרסם בכתב העת Mind, כמדד אפשרי למידה שבה יש למכונה כלשהי אינטליגנציה. במקור טיורינג כינה מבחן זה בשם "משחק החיקוי" (The Imitation Game). המבחן נעשה בדרך הבאה: חוקר מקיים דיאלוג בשפה טבעית עם שני גורמים סמויים מעיניו, האחד אדם והשני מכונה. אם החוקר אינו מסוגל לקבוע בביטחון מי האדם ומי המכונה, אזי המכונה עברה בהצלחה את המבחן. כדי לפשט את המבחן, הוא נערך בערוץ תקשורת טקסטואלי בלבד. מאז טיורינג ועד היום מדענים רבים מתעסקים באובסיביות בנושא מרתק זה. אפשר אפילו לומר ביומרה כי תחום זה הוא פסגת שאיפותיו של כל מתכנת מחשבים באשר הוא. בשנים האחרונות עם התפתחות המחקר והפיתוח בנושא הבינה המלאכותית בכלל ולמידת המכונה בפרט, אנו מוצאים כי פתרון בעיות מורכבות שעד כה נעשו בידי אדם עובר בהדרגה לתחום הבעיות שניתנות לפתרון ביד מחשב.

ניתן לכלול בתחום של בינה מלאכותית שני נושאים היררכיים:

1. **למידת מכונה (Machine Learning)** - פיתוח אלגוריתמים המיועדים לאפשר למחשב ללמוד מתוך דוגמאות, ולפעול במגוון משימות חישוביות בהן התכנות הקלאסי איננו אפשרי. אין לבלבל בין תחום זה, שבו המחשב הוא הלומד, ובין למידה ממוחשבת, שבה המחשב משמש כעזר ללמידה על ידי הרצת לומדה או בדרך אחרת.
2. **למידה עמוקה(Deep Learning)** - תחום זה הוא תת תחום בלמידת מכונה כך שהוא מתמקד בעיקר ברשתות נוירונים בעלות מספר רב של שכבות.

בין הנושאים "החמים" היום בתחום הלמידה העמוקה ניתן למצוא את תחום - "**ניתוח שפה טבעית**" (NLP-Natural Language Processing) בו מתבצע בנתונים עיבוד וניתוח בעיקר על קלטי טקסט ושפה. עד כה הפיתוח והמחקר העיקרי שנעשה בנושא עסק בשפות בהן המידע רב יותר ונגיש יותר כמו אנגלית צרפתית וכדו', אולם המחקר בנושא ה**שפות שמיות** כמו עברית וערבית נזנח וההתעסקות בו קטנה יחסית. בפרויקט זה לקחנו על עצמנו לבצע ניתוח של טקסטים בעברית בכוונה על מנת להגביר את העיסוק בנושא עם הבנת חשיבות הידע לצרכים יומיומיים בשפה העברית וכן חשיבות ביטחונית כאשר מדובר על שפות שמיות כמו ערבית פרסית וכדו'.

1.3 מטרת המחקר ומהלך העבודה

נתאר בקצרה את מטרות המחקר ומהלך העבודה שלנו על ארבעת שלביה:

1. עיבוד הטקסטים בעברית לווקטורים - בשלב הראשון לקחנו 50000 משפטים בעברית, ביצענו סינון והסרנו את המשפטים ללא הנושא.
2. השתמשנו בתייגן ותייגנו באופן מתאים כל משפט וסימנו בו את הנושא.
3. עיבוד מקדים (Pre-Training) - העיבוד התבצע בעזרת שיטת המרה לווקטורים הנקראת word2vec שיטה מוכרת בתחום ה-NLP והוכחה כיעילה במיוחד למטרות הפרויקט.
4. בניית רשת ניתוח מתאימה לדרישות - הניתוח נעשה בעזרת רשת נוירונים RNN משלושה סוגים: LSTM, Bi-LSTM, GRU רשתות מהסוג הזה הוכיחו את עצמן במיוחד בשימושים לצורכי NLP וכדו'. כדי לייעל את הניתוח נשתמש במכניקה של Attention. שיטות אלה יעזרו למקסם את היכולות של הרשת וכן להפוך אותה לשימושית להרבה סוגים של טקסטים.
5. הרצת המשפטים על גבי הרשת ויצירת מודל מתאים לחיזוי נושא במשפט.
6. הסקת מסקנות מחקריות ומחשבה על ייעול עתידי.

2 שפה טבעית

נתונים טקסטואליים הם נתונים לא מובנים (unstructured). לרוב הם שייכים לשפה ספציפית בעקבות תחביר וסמנטיקה השייכים גם הם לאותה שפה. כל פיסת נתוני טקסט - מילה, משפט או מסמך – מתייחסת ברוב הזמן לשפה טבעית כלשהי. בחלק זה נבחן את ההגדרות של שפה טבעית, פילוסופיה של שפה, רכישת שפה ושימוש בשפה. כדי להבין כיצד לנתח טקסטים וטכניקות עיבוד שפה טבעית, עלינו להבין מה הופך את השפה ל"טבעית".

מהי שפה טבעית? במילים פשוטות, שפה טבעית היא שפה שפותחה והתפתחה על ידי בני אדם באמצעות שימוש טבעי בתקשורת, ולא בנויה ונוצרה באופן מלאכותי (כמו שפות תכנות מחשבים). שפות אנושיות כמו עברית, ויפנית הן שפות טבעיות. ניתן לתקשר בין שפות טבעיות בצורות שונות, כמו דיבור, כתיבה, או אפילו סימנים. כהקדמה הכרחית למחקר שלנו ננסה להבין את המקורות, הטבע והפילוסופיה של השפה ומתוכם לגזור עקרונות מנחים בעבודה עם שפות טבעיות.

2.1 פילוסופיה של שפה

אחרי שראינו מהי שפה טבעית, חשוב לשאול את השאלות הבאות:

- מהם מקורות השפה?
- מה הופך את השפה העברית ל"עברית"?
- כיצד יש משמעות למילה "ארטיק"?
- כיצד בני אדם מתקשרים בינם לבין עצמם באמצעות השפה?

שאלות אלה ועוד רבות עומדות בבסיס כל שפה. הפילוסופיה של השפה עוסקת בעיקר בארבע הבעיות הבאות ומחפשת תשובות לפתרונן.

1. אופי המשמעות של המילים בשפה.
2. אופי השימוש בשפה.
3. הכרה/ קוגניציה (cognition) בשפה.
4. הקשר בין שפה למציאות.

כעת נבחן כל אחת מהשאלות הללו. ההתעסקות בשאלות הללו אולי נראות מיותרות, אולם על ידם נוכל לגזור עקרונות חשובים להמשך המחקר.

אופי המשמעות של המילים בשפה

שאלת אופי המשמעות בשפה עוסקת בסמנטיקה של השפה ואופי המושג 'משמעות' בפני עצמו. במקרים אלה פילוסופים של שפה או בלשנים מנסים לברר מה המשמעות של "בעצם" להתכוון לכל דבר - כלומר, איך ומהיכן משמעותה של כל מילה או משפט התחילה וממשיכה להיווצר ואיך בכלל מילים שונות בשפה יכולות להיות מילים נרדפות זו לזו?. כיצד מילים שונות מייצרות יחסים בין אחת לרעותה?. דבר נוסף שיש לו חשיבות כאן הוא

כיצד מבנה ותחביר בשפה סוללים את הדרך לסמנטיקה? אם נהיה ספציפיים יותר: כיצד מלים, שיש להן משמעויות משלהן, גורמות יחד ליצירת משפטים משמעותיים בעלי מובנות מורכבת?. **בלשנות** היא המחקר המדעי של השפה. זהו תחום מיוחד העוסק בכמה מהבעיות הללו. תחביר, סמנטיקה, דקדוק ועצי ניתוח הם רק כמה דרכים לפתור את הבעיות האלה. אופי המשמעות יכול להתבטא בבלשנות בין שני בני אדם, ובמיוחד בין מדבר למקשיב/קולט (receiver & sender), אם נסתכל על מה שהאדם שמדבר מנסה לבטא או לתקשר כשהוא שולח הודעה למקלט, ומה שהמקלט בסופו של דבר מבין ומסיק מההקשר של ההודעה שהתקבלה. גם מבחינה לא לשונית, דברים כמו שפת גוף, התנסויות קודמות והשפעות פסיכולוגיות תורמים למשמעות של השפה, כאשר כל בן אנוש תופס או מביא משמעות בדרכו שלו, תוך התחשבות בכמה מהגורמים הנ"ל.

אופי השימוש בשפה

המחקר בנוגע לשימוש בשפה עוסק יותר באופן ובצורה בה משתמשים בשפה כישות ואמצעי בתרחישים שונים ובתקשורת בין בני אדם. מחקר זה כולל ניתוח הדיבור והשימוש בשפה בזמן הדיבור, כולל כוונת הדובר, הטון, התוכן והפעולות הכרוכות בהבעת דיבור. מה שמכונה בבלשנות לעתים קרובות "**מעשה דיבור**". מושגים מתקדמים יותר כמו 'מקורות יצירת השפה' ו'פעילויות קוגניטיביות אנושיות' האחראיות על לימוד ושיח שמתבצעות בהקשר של רכישת שפה הם גם חלק עיקרי בתחום זה.

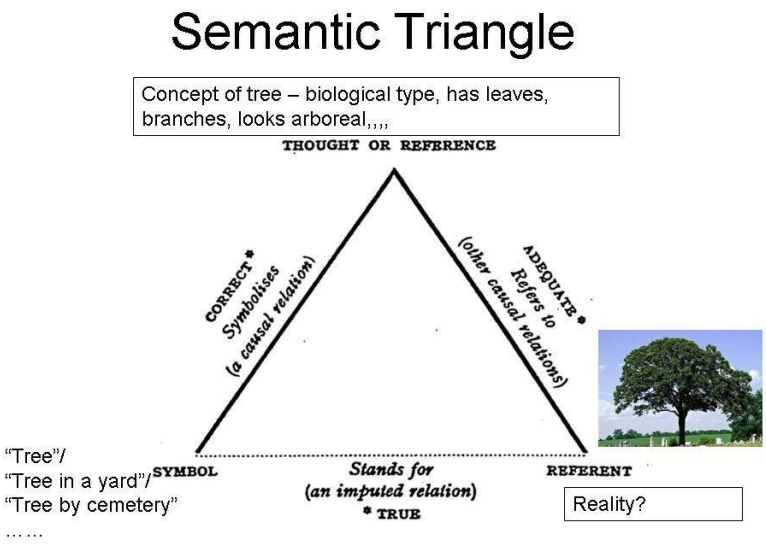
הכרה/קוגניציה (cognition) בשפה

קוגניציה של שפה מתמקדת באופן ספציפי באופן בו הפונקציות הקוגניטיביות של המוח האנושי אחראיות לפרשן ולהבין את השפה. בהתחשב בדוגמה של שולח ומקבל טיפוסי, ישנן פעולות רבות הקשורות לדרך מסירת ההודעה ולפרשנות שלה. קוגניציה מנסה לגלות כיצד המוח מגיב ועובד בשילוב והקשר של מילים ספציפיות למשפטים ומשם הפיכת המילה למסר משמעותי ומה הקשר של השפה לתהליך המחשבה של השולח והמקבל בזמן שהם משתמשים בשפה כדי להעביר מסרים.

הקשר בין שפה למציאות

הקשר בין שפה למציאות בוחן את מידת האמת של ביטויים שמקורם בשפה. בדרך כלל, פילוסופים של השפה מנסים למדוד עד כמה לביטויים הללו יש אחיזה במציאות כלומר עד כמה הם מתארים עובדה. כמו כן הפילוסופים יתעסקו בשאלה כיצד המילים קשורים לתרחישים מסוימים בעולמנו. קשר זה יכול לבוא לידי ביטוי בכמה אופנים, בהמשך ננסה לחקור כמה מהם.

משולש ההתייחסות הוא אחד הדגמים הפופולאריים ביותר המשמש להסביר כיצד מילים מעבירות משמעות ורעיונות במוחו של המקבל וכיצד משמעות זו קשורה בחזרה לישות או

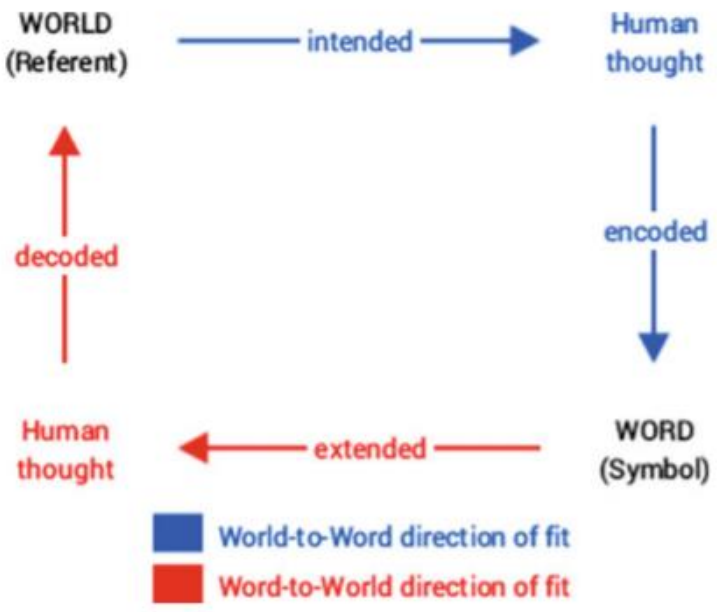


איור 1משולש ההתייחסות

עובדה עולמית אמיתית. משולש ההתייחסות הוצע על ידי צ׳רלס אוגדן ואיבר ריצ׳רדס בספרם, "משמעות המשמעות"[1], שפורסם לראשונה בשנת 1923, ומסומן ב**איור 1.**

המשולש של מודל ההתייחסות המוצג ב**איור 1** ידוע גם כ׳משמעות של מודל המשמעות׳, כפי שמוצג באיור 1 בו נמצאת דוגמה אמיתית לעץ שנתפס על ידי אדם שנמצא מולו. **סמל** מצוין כסמל לשוני, כמו מילה או חפץ שמעורר מחשבה במוחו של אדם. במקרה זה, הסמל הוא העץ, וזה מעורר מחשבות כמו - *מה זה עץ, אובייקט מעולם הצומח שמביא לעיתים פירות, או סמל של חטיבת גולני לדוגמה.* מחשבות אלו ידועות כהפניה (מעתה **׳רפרנס׳** reference) ובאמצעות התייחסות זו האדם מסוגל *לקשר* בין המילה עצמה למשהו שקיים בעולם האמתי, המכונה **רפרנט**. במקרה זה הרפרנס הוא העץ שהאדם רואה בזמן שהוא נוכח מולו. הדרך השנייה לגלות קשרים בין שפה למציאות ידועה *ככיוון של התאמה* (direction of fit). נדבר כאן על שני כיוונים עיקריים. כיוון ההתאמה הראשון נקרא *התאמת מילה ← עולם* (word-to-world) מדבר על מקרים שבהם השימוש בשפה יכול לשקף את המציאות. כיוון זה מצביע על שימוש במילים בכדי להתאים או להתייחס למשהו שקורה או שכבר קרה בעולם האמיתי. דוגמא לכך היא המשפט *"גשר המיתרים מאוד גבוה"*, שמדגיש עובדה במציאות. כיוון ההתאמה השני מכונה *התאמת עולם ← מילה* (world-to-word), מדבר על מקרים שבהם השימוש בשפה יכול לשנות את המציאות. דוגמה כאן היא המשפט *"אני מתכנן להרזות"*, במקרה זה האדם הוא זה שמשנה את המציאות על ידי הרזיה אותה ייצג במשפט שמועבר. **איור 2** מציג את הקשר בין שני כיווני ההתאמות.

[1] 1923 The Meaning of Meaning



איור 2 הקשר בין שני כיווני התאמות

די ברור מ**איור 2** כי בהתבסס על הרפרנס הנתפס מהעולם האמיתי, אדם יכול ליצור ייצוג בצורה של סמל או מילה וכתוצאה מכך יכול לתקשר ולהעביר את הייצוג הזה אל אדם אחר כך שאותו אדם עובר מן המילה אל הייצוג ומצליח להבין את המשמעות של המילים וכך זה ממשיך בצורה מחזורית.

2.2 בלשנות

אחרי שעסקנו קצת עם המשמעות של שפה טבעית - כיצד היא נלמדת ומשמשת כדרך תקשורת בין בני אדם נוכל להבין את המושג '**בלשנות**'. תחת כותרת זו ניתן להכניס את כל השאלות שהתעסקנו בהם ועוד נתעסק. אנשי המחקר בתחום זה נקראים *בלשנים*. באופן רשמי, בלשנות מוגדרת כלימוד המדעי של השפה, כולל צורה ותחביר של שפה, משמעות וסמנטיקה המתוארת על ידי השימוש בשפה ובהקשר שבו משתמשים בשפה. ניתן לתארך את מקורותיה של הבלשנות למאה הרביעית לפני הספירה, כאשר המלומד והבלשן ההודי פאניני (Panini) הפך את התיאור של השפה *סנסקריט* לביטוי רשמי. המונח בלשנות הוגדר לראשונה בשנת 1847, בכדי לציין את המחקר המדעי אודות שפות. לפני כן שימש המונח פילולוגיה כדי לציין את אותו הדבר. למרות שאין צורך בחקר בלשני מפורט בשביל לנתח טקסטים, כדאי לדעת את התחומים השונים בהם עוסקת הבלשנות מכיוון שחלקם באים לעיתים קרובות לידי ביטוי בעיבוד שפות טבעיות ובאלגוריתמים לניתוח טקסטים.

התחומים העיקריים המובחנים במחקר הבלשנות הם כדלקמן:

- **פונטיקה** (Phonetics): ענף המחקר העוסק בתכונות האקוסטיות של צלילים המיוצרים על ידי מערכת הקול האנושית במהלך הדיבור. תחום זה כולל לימוד הצלילים וכן כיצד צלילים נוצרים על ידי בני אדם. היחידה האישית הקטנה ביותר של

דיבור אנושי בשפה ספציפית נקראת פונמה[2] (phoneme). מונח כללי יותר בשפות עבור יחידת דיבור זו הוא פון (phone).

- **פונולוגיה** (Phonology): זהו חקר דפוסי הקול כפי שהם מתפרשים במוח האנושי ומשמשים להבחנה בין פונמות שונות כדי לגלות אילו מהן משמעותיות. המבנה, השילוב והפרשנויות של פונמות נלמדים בפירוט, בדרך כלל על ידי התחשבות בשפה מסוימת בכל פעם. בעברית הקלאסית יש כ-30 פונמות, רובן מסומנות בכתיבה מנוקדת. פונולוגיה בדרך כלל נרחבת מעבר ללימוד פונטיות בלבד וכוללת דברים כמו מבטאים, טון, משקלים והברה.
- **תחביר** (Syntax): לרוב מדובר בחקר משפטים, ביטויים, מילים והבניינים שלהם. זה כולל לחקור כיצד מילים משולבות יחד דקדוקית ליצירת ביטויים ומשפטים. סדר תחבירי של מילים המשמשות בביטוי או משפט משפיע מאוד מכיוון שהסדר יכול לשנות את המשמעות של המשפט לחלוטין.
- **סמנטיקה** (Semantics): בענף זה עוסקים בחקר המשמעויות של המילים בשפה. ניתן לחלק אותו לסמנטיקה לקסיקלית וקומפוזיציות.
  - **סמנטיקה לקסיקלית (Lexical semantics):** חקר המשמעות של מילים וסמלים באמצעות מורפולוגיה ותחביר.
  - **סמנטיקה קומפוזיציונית (Compositional semantics)**: לימוד מערכות יחסים בין מילים, שילובי מילים והבנת המשמעויות של ביטויים משפטים וההקשר ביניהם.
- **מורפולוגיה** (Morphology): מורפמה (morpheme) בעברית **צוּרָן** היא היחידה הלשונית הקטנה ביותר הנושאת משמעות. זאת להבדיל מפונמה ,שהיא היחידה הקטנה ביותר העשויה להשפיע על המשמעות, אך אין לה משמעות משל עצמה. ענף זה כולל מילים, קידומות, סיומות וכדומה שיש להם משמעות מובהקת משל עצמם. מורפולוגיה היא עבודת לימוד המבנה והמשמעות של יחידות או צורנים ייחודיים בשפה. כללים ותחבירים ספציפיים שולטים בדרך כלל באופן שבו מורפמות יכולות להשתלב יחד.
- **לקסיקון** (Lexicon): ענף העוסק בחקר התכונות של מילים וביטויים המשמשים בשפה וכיצד אוצר המילים של השפה נבנה בעזרתם. חקר זה כולל גם את השאלות, אילו סוגים של צלילים במילה קשורים למשמעות של המילה, חלקי הדיבור שמילים שייכים להם, וצורותיהם המורפולוגיות.
- **פרגמטיקה** (Pragmatics): נושא המחקר כיצד גורמים לשוניים וגם לא לשוניים כמו הקשר ותרחיש, עשויים להשפיע על המשמעות של ביטוי של הודעה או אמירה. ענף זה כולל גם את השאלה האם יש משמעות נסתרת או עקיפה בתקשורת.
- **ניתוח שיחה** (Discourse analysis): ענף של ניתוח שפה והחלפת מידע בצורה של משפטים על פני שיחות בין בני אדם. כלומר האבחנה בין שיחות המועברות בדרכים שונות כמו לדבר, לכתוב או אפילו לחתום.

---

[2] לפי הצעת האקדמיה ללשון העברית: 'הֶגְיָן'

- **סיגנון** (Stylistics): חקר השפה תוך התמקדות בסגנון של הכתיבה, הטון, המבטא, הדיאלוג, הדקדוק וסוג הקול.
- **סמיוטיקה** (Semiotics): לימוד של סימנים, סמלים ותהליכי סימן וכיצד הם מעבירים משמעות. דברים כמו אנלוגיה, מטאפורות וסמליות נמצאים בתחום זה.

למרות שהתחומים לעיל הם תחומי הלימוד והמחקר העיקריים, בלשנות הוא תחום עצום עם היקף גדול בהרבה ממה שמוזכר כאן. עם זאת, דברים כמו תחביר, שפה וסמנטיקה הם חלק מהמושגים החשובים ביותר שלרוב מהווים את היסודות לעיבוד שפה טבעית. בחלק הבא נבחן אותם מקרוב.

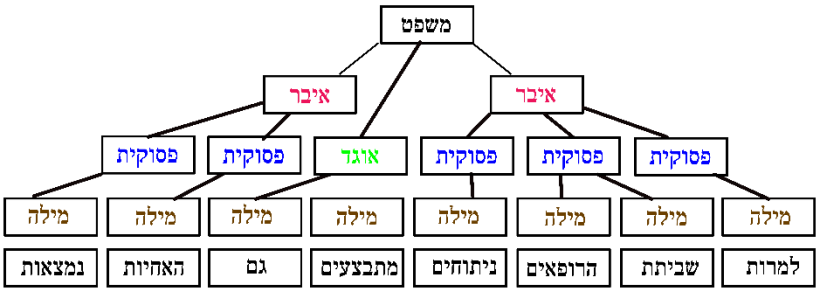
2.3 תחביר ומבנים בשפה

אחרי שאנחנו יודעים קצת על המשמעות של שפה ועל מבנים בשפה אנחנו יכולים להבין מדוע מבנים בשפה ותחביר הולכים יד ביד. בחלק זה נדבר ספציפית על התחביר והמבנה בשפה העברית מכיוון שבמחקר זה נעסוק בנתונים טקסטואליים השייכים לשפה העברית. אך ניתן להרחיב גם הרבה מהמושגים הללו לשפות אחרות (במיוחד לשפות שמיות). ידע על מבנים ותחביר של שפה מועיל בתחומים רבים כמו עיבוד טקסט, ביאור וניתוח לצורך פעולות נוספות כגון סיווג טקסט או סיכום של טקסטים. בעברית, מילים בדרך כלל משתלבות יחד ליצירת יחידות מורכבות אחרות. יחידות אלה כוללות מילים, ביטויים, סעיפים ומשפטים. כל המרכיבים הללו קיימים יחד בכל הודעה והם קשורים זה לזה במבנה היררכי. יתר על כן, משפט הוא פורמט מובנה של ייצוג אוסף מילים בתנאי שהם מצייתים לכללים תחביריים מסוימים כמו דקדוק. התבונן בקבוצת המילים המיוצגת **באיור 3.**

מאוסף המילים **באיור 3**, קשה מאוד לברר מה הם מנסים להעביר או להתכוון. אכן, השפות אינן מורכבות רק מקבוצות של מילים. משפטים עם תחביר ראוי לא רק עוזרים לנו לתת מבנה נכון ולקשר מילים יחד, אלא גם עוזרים לנו להעביר משמעות על סמך סדר המילים או מיקום במשפט. בהתחשב בהיררכיה של המשפט → איבר → פסוקית → מילה, אנו יכולים לבנות את עץ המשפט ההיררכי המוצג **באיור 4** באמצעות 'ניתוח רדוד' – טכניקה המשמשת לבדיקת חלקי המשפט

ניתוחים נמצאות למרות גם הרופאים
מתבצעים שביתת האחיות חשובים

איור 3 מילים ללא סדר



איור 4 עץ היררכי

מהעץ ההיררכי **באיור 4**, אנו מקבלים את המשפט *למרות שביתת הרופאים ניתוחים מתבצעים וגם האחיות נמצאות*. אנו יכולים לראות שהעלים של העץ מורכבים ממילים, שהן היחידה הקטנה ביותר כאן, ושילובי מילים יוצרים פסוקיות, אשר בתורם יוצרים איברים. איברים מחוברים זה לזה דרך מילות קישור וניגוד, אוגדים ועוד. בצורה זו הם יוצרים יחד את המשפט הסופי. בחלק הבא, נסקור את כל המרכיבים ונברר את הקטגוריות התחביריות העיקריות בפירוט רב יותר ונבין כיצד לנתח אותם.

מילים

מילים הן היחידות הקטנות ביותר בשפה שהן עצמאיות ובעלות משמעות משלהן. למרות שמורפמות הן היחידות המובחנות הקטנות ביותר, מורפמות אינן עצמאיות כמו מילים, ומילה יכולה להיות מורכבת מכמה מורפמות. מומלץ לתייג מילים ולנתח אותן לחלקי הדיבור שלהן (POS) בכדי לראות את הקטגוריות התחביריות העיקריות. כאן, נסקור את הקטגוריות העיקריות והמשמעות של תגי ה- POS השונים. בהמשך נבחן אותם בפירוט רב יותר ונבחן את השיטות לחיזוי מילים באופן נכון.

בדרך כלל מילים יכולות להיכנס לאחת הקטגוריות הבאות:

- **שם עצם**: מציין מילים המתארות אובייקט או ישות כלשהם שעשויים להיות חיים או שאינם חיים. כמו בדוגמא **באיור 4** "גשם". סמל תג ה- POS לשמות עצם הוא **N**.
- **פעלים**: מילים המשמשות לתיאור פעולות, מצבים או התרחשויות מסוימות. יש מגוון רחב של קטגוריות משנה נוספות, כמו פעלי עזר ורבים אחרים. כמה דוגמאות אופייניות לפעלים הם – *רץ, קפץ, כתב שובתים*. סמל תג ה- POS עבור פעלים הוא **V**.
- **שמות תואר:** מילים המשמשות לתיאור או הכשרה של מילים אחרות, בדרך כלל שמות עצם וביטוי עצם. לביטוי "פרח יפה" יש את שם העצם (N) פרח שמתואר או מוסמך באמצעות התואר (ADJ) יפה. סמל תג ה- POS לתארים הוא **ADJ**.

מילת קישור	שם עצם	שם עצם	שם עצם	פועל	מילת קישור	שם עצם	פועל
למרות	שביתת	הרופאים	ניתוחים	מתבצעים	גם	האחיות	נמצאות

איור 5 תיוג POS

- **תואר הפועל** (Adverb): היא מילה שמשנה פועל ,שם תואר ,איבר , מילת יחס או משפט. תואר הפועל בדרך כלל מספק מידע אודות האופן, המקום, הזמן, התדירות, התואר, רמת הוודאות, וכד', בכדי לענות על שאלות כגון איך?, מתי?, איפה?, כמה פעמים?, ובאיזה אופן?. מילים כמו "לאט", "בקול רם", "בזהירות", "במהירות", "בשקט" או "בעצב" הם כולם תוארי פועל. הפרח יפה. סמל תג ה-POS לתואר הפועל הוא **ADV**.

מלבד ארבע הקטגוריות העיקריות הללו של חלקי דיבור, ישנן קטגוריות אחרות המופיעות לעיתים קרובות בשפה העברית. אלה כוללים כינויי-שם, מילות-יחס, מילות קישור, צירופים ורבים אחרים. יתר על כן, כל חלוקת POS כמו שם העצם (N) ניתנת לחלוקה נוספת לקטגוריות כמו שמות עצם יחידים (NN), שמות עצם עבור יחידים (NNP) ושמות עצם עבור רבים (NNS).

באיור 5 ניתן לראות את הדוגמא הקודמת שלנו לאחר שנוסיף לה תגי POS. ניתן להבחין במספר תגים חדשים כמו מילת היחס *למרות* שמביע ניגוד וכן מילת הקישור *גם* המביעה חיבור בין שני תתי המשפט

פסוקיות

מונח המתייחס להיגדים מילוליים מוגדרים. המונח פסוקית משמש במספר דיסציפלינות .הוראתו היסודית, המשותפת כמעט לכל הדיסציפלינות, היא פסוק משנֶה ('משפט משנה'); חלקו של פסוק ('חלקו של משפט'); פסוק קטן ('משפט קטן'). המילה 'פסוקית' היא תוצאת הקטנה על פי גזירה קווית של בסיס + סופית של הצורן הסופיx.ית – 'פסוקית', פסוק קטן ומשני ביחס לפסוק הרגיל. בתורת התחביר, ה**פסוקית** מתארת את תפקיד החלקים שבמשפט המשועבד . לדוגמה: במשפט "ידוע ש-העישון מזיק לבריאות", פסוקית הנושא היא "העישון מזיק לבריאות ".

למשפט המשועבד קיימות מספר פסוקיות: פסוקית נושא; פסוקית נשוא; פסוקית מושא; פסוקית תיאור (פסוקית תיאור כוללת בתוכה: פסוקית זמן; פסוקית מקום; פסוקית מידה; פסוקית סיבה; פסוקית תכלית; פסוקית תנאי; פסוקית ויתור; פסוקית מצב; פסוקית אופן); פסוקית לוואי. במחקר זה נתייחס **לפסוקית הנושא** באופן מפורט יותר בהמשך.

2.4 דקדוק

דקדוק הוא הכלי שבעזרתו ניתן להבנות בשפה **תחביר**. דקדוק מורכב בד"כ ממערכת כללים המשמשים לקביעת אופן הסידור של מילים כאשר נרצה לבנות משפט בעל משמעות. הדקדוק איננו מוגבל למילה הכתובה בהכרח, הוא גם מופיע כאשר אנחנו משוחחים בע"פ. כללי הדקדוק יכולים להיות ספציפיים לשפה מסוימת אבל גם יכולים להיות אוניברסליים למספר שפות (ראה להלן הדקדוק האוניברסלי). במקרה של המחקר שלנו אנחנו מתעסקים עם אלמנטים של שפה אוניברסליים הנקראים מודל **SVO** (Subject-Verb-Object) נושא – פועל – מושא. מודל זה מגדיר את היחסים בין חלקי המשפט העיקריים. ההיסטוריה של הדקדוק עשירה מאוד וכוללת מקורות רבים, שכבר החלו מסנסקריט בהודו. בהתייחס לתרבות המערב, מחקר הדקדוק מקורו ביוונים, והעבודה המוקדמת ביותר הייתה 'אומנות הדקדוק', שנכתבה על ידי דיוניסיוס תראקס (Dionysius Thrax). דגמי הדקדוק הלטיני פותחו מהמודלים שפותחו ע"י היוונים, ובהדרגה לאורך כמה דורות נוצרו דקדוקים לשפות שונות. רק במאה ה -18 נחשב הדקדוק כמועמד רציני להיות תחום עצמאי תחת המקצוע בלשנות. הדקדוקים התפתחו במהלך הזמן, מה שהוביל ללידת סוגים חדשים של דקדוקים, ודקדוקים ישנים איבדו אט אט את המקום שלהן לטובת דקדוקים חדשים. מכאן אנחנו יכולים להבין שמערכת הדקדוק איננה מערכת קבועה, אלא מערכת שמתפתחת ומשתנה עם הזמן. כמובן שהשינויים נובעים מהצורות בהן בני האדם משתמשים בשפה. כלומר המציאות מבנה את השפה ואת החוקים שבה.

בשפה העברית מקובל לחלק את המילים במשפט לשתי נקודות מבט כמופיע ב**איור 6.**

1. **חלקי דיבר** - חלוקה לחלקי דיבר, או במינוח מקצועי יותר לקטגוריות לקסיקליות, היא סיווג כל מילה בלקסיקון של שפה לפי מאפיינים מורפולוגיים.
2. **חלקי המשפט (הפשוט)** - אפשר לקבוע לאיזה חלק דיבור מילה שייכת גם בלי לשבץ אותה בתוך משפט. הקביעה נעשית לפי תכונות קבועות של המילה. אך בלי משפט אי-אפשר לקבוע למילה תפקיד תחבירי. "נושא", "נשוא", "מושא" קיימים רק אם יש משפטים. הסביבה של המשפט יוצרת תפקיד תחבירי. הדבר דומה ליחס בין השם הפרטי של האדם לבין תפקידיו. לדוגמה: בחורה בשם "מיכל", אשר יש

	חלקי דיבור	חלקי המשפט
דוגמות	פועל, שם עצם, תואר, תואר הפועל, שם מספר, מילות יחס	נשוא, תיאור מקום, תמורה, לוואי, הסגר, מושא, נושא
הקביעה לפי	ניתוח מילים תכונות נסייה של המילה **תכונה קבועה** שולחן = שם עצם	ניתוח משפטים תפקיד המילה במשפט **תפקיד משתנה לפי המשפט** השולחן התפרק. שולחן=נושא למגירות השולחן יש ידיות מעץ. שולחן=לוואי עמדתי ליד השולחן. שולחן= ת' מקום
ניתוח	**חלקי דיבור** ש"ע הנגר, פועל בנה, ש"ע שולחן.	**חלקי משפט** נושא הנגר, נשוא קנה, מושא שולחן.

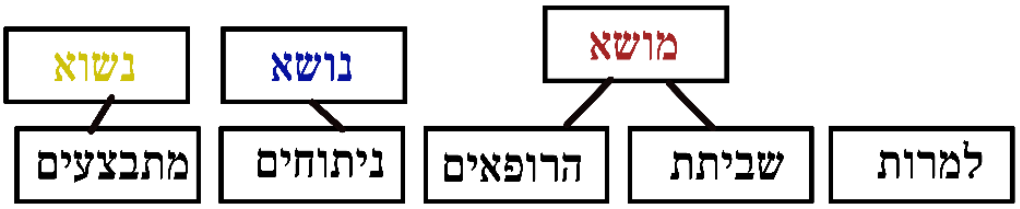
איור 6 טבלת ניתוח חלקי דיבר וחלקי משפט

לה תכונות קבועות כמו גובה, מבנה גוף, צבע עור, יכולה בזמנים שונים למלא תפקידים שונים לפי הסביבה המשתנה: חברה, אם, מורה, בת, רעיה, זמרת ועוד.

חלקי הדיבר הנפוצים ביותר הם שם העצם והפועל, ולצידם קיימים שמות תואר, תארי פועל, כינויים, מילות יחס, מילות קישור ועוד. בעבר היה נהוג לחלק את חלקי הדיבר לפי קריטריונים סמנטיים: שם עצם הוגדר כ"מילה המציינת חפץ או מושג", ופועל הוגדר כ"מילה המציינת פעולה". ואולם אפיונים אלו כלל אינם מדויקים, והם גם קשים מאוד להגדרה. הבלשנות המודרנית מעדיפה הגדרות המבוססות על מאפיינים מורפולוגיים ותחביריים: *'שם עצם'* - צורה הנוטה לפי מספר (יחיד, רבים, זוגי) ולפעמים גם לפי מין (זכר, נקבה, סתמי), ויכולה לשמש כנושא או מושא במשפט. דוגמאות: "בית" (שם עצם כללי), "ירושלים" (שם עצם פרטי). *'שם פעולה'* - שם המגדיר את הפעולה. דוגמאות לשמות פעולה: "אכילה" "ריקוד" וכו'. *'פועל'* - צורה הנוטה (בדרך כלל) לפי פעולה, גוף וזמן לדוגמה: כתבנו - כתיבה, אנחנו, עבר. פועל על פי רוב משמש כנשוא במשפט ובחלק מהמקרים מקבל מושא. דוגמאות: "הלכתי", "עומד". 'שם פועל' - שייך למערכת הפועל, אך אין לו זמן. אי אפשר ליידע שם פועל, ותמיד יכלול את מילית היחס **'ל'**. דוגמאות: "ללמוד", "לאכול", "לרקוד". 'שם תואר' - צורה המשמשת כלוואי לשם עצם או כנשוא במשפט שמני. נוטה בדרך כלל לפי מין ומספר. דוגמאות: "גדולה", "חופשיים". 'תואר הפועל' - תואר שמתאר את הפועל לדוגמה: "הוא רץ מהר" מהר הוא תואר הפועל (המתאר את אופי הריצה).

חלקי המשפט/תחביר התחביר מגדיר איזה רצף של מילים מהווה משפט נכון ותקין. בעברית, בשונה משפות אחרות, לכל חלק במשפט יש תפקיד ושם.

התחביר איננו מתמקד ברכיבים עצמם כמו פסוקיות משפטים. הדגש בתחביר הוא על ה'תפקיד' של המילה במשפט. התפקיד של המילה מגדיר את היחס שלה אל שאר המילים במשפט. כדי להבין את חלקי המשפט, עלינו לדעת תחילה מה פירוש היחסים בהקשר הזה. לכל מילה יש הקשר או שהיא תלויה במילה אחרת המבוססת על מיקום המילים במשפט. כתוצאה מכך, דקדוקי היחס מניחים שמרכיבים נוספים של ביטויים וסעיפים נגזרים מהמבנה התלותי הזה בין המילים. העיקרון הבסיסי העומד מאחורי דקדוק תלות הוא שבכל משפט בשפה, לכל המילים, למעט מילה אחת, יש קשר או תלות כלשהי במילים אחרות במשפט. המילה שאין לה תלות נקראת נשוא, הנשוא הוא החלק ההכרחי בכל משפט ולכן הוא השורש. בד"כ נשוא הוא פעולי אבל לא תמיד. כל המילים האחרות מקושרות ישירות או בעקיפין לפועל השורש באמצעות יחסים של נושא, של הנשוא, וכן המושא, שהם אלה שמתארים את התלות.



איור 7 עץ תחביר

כמו בכל מבנה במשפט ניתן להציג גם את חלקי המשפט באמצעות יער ניתוח המופיע **באיור 7.**

המשפט הפשוט

המשפט הפשוט בהגדרה הוא משפט בו יש **נשוא** אחד, שאר המשפטים הם צירופים של משפטים פשוטים.

- **נשוא** - הנשוא הוא לרוב רכיב המשפט המהווה חידוש, אך על פי רוב זיהויו מתבסס דווקא על קריטריון צורני: במשפט פועלי הנשוא הוא הפועל. מאליו ברור, כי זיהוי כזה איננו אפשרי במשפט שמני. מבחינה תחבירית, הנשוא נחשב בבלשנות המודרנית לגרעין המשפט, כלומר הרכיב נשואו של משפט פועלי הוא בהכרח הפועל. בבלשנות המודרנית, הנשוא נתפס כגרעין המשפט, כלומר הרכיב אשר "שולט" בשאר חלקי המשפט בהיותם הרחבותיו (בין אם כמשלימים מושאיים או תיאוריים, או נושא המשפט). הנשוא נחשב גם ללב הסמנטי של המשפט: באסכולות מסוימות של הבלשנות, הוא נתפס כפרדיקט לוגי אשר ניתן לייחס לו ערך אמת (כלומר "אמת" או "שקר").

- **נושא** - נושא (סובייקט) הוא תואר שמציין את תפקידה התחבירי של מילה (או מספר מילים) במשפט שאותו מאפיין נשוא המשפט. יש לשים לב שיש שתי הגדרות למונח נושא. מבחינה תחבירית ומבחינת הרעיון המרכזי של משפט (topic). מושג הנושא משלב בתוכו במסורת הבלשנית שלושה אפיונים שונים. אפיון לפי מבנה המסר: הנושא הוא התמה של המשפט. כלומר, הנושא הוא הרכיב הידוע, ש"עליו מדבר המשפט". יש כאלה אשר משתמשים במושג 'נושא לוגי' לתיאור אפיון זה.

בעברית קל לאפיין את נושא המשפט הפועלי, כרכיב במשפט אשר גורר התאם עם הפועל[3]:

הילד אכל תפוח.

הילדה אכלה תפוח.

במשפט השמני המצב סבוך יותר. על פי רוב, הנושא מאופיין בהיותו הרכיב המיודע של המשפט, בעוד שהנשוא הוא הרכיב הבלתי מיודע:

הילד גבוה.

המלך הוא עירום.

אפיון זה בעייתי במשפטים שמניים בהם שני הרכיבים הם מיודעים או בלתי-מיודעים:

ילדים זה שמחה.

הנשיא הוא ראש המדינה.

בפועל, זיהוי הנושא במשפטים שמניים נעשה לפי מבנה המסר: הרכיב הידוע, ש"עליו מדבר המשפט" מאופיין כנושא. במשפטים אחרים, זיהוי הנושא מתבסס בעיקרו על קונבנציה תחבירית:

יש לי **כסף**.

אסור **לעשן**.

- **מושא** - בתורת התחביר, המושא משמש כהשלמה לנשוא הפעולי, הקישור לפועל הוא מחייב והפועל נזקק להשלמתו של המושא. השלמה זו באה תמיד אחרי מילת יחס או מספר מילות יחס, כך שמילות אלה מצטרפות לפועל והן חלק מילוני של הפועל. לדוגמא:

הבחין ב, האמין ל, השלים עם, אכל את.

את מילת היחס המוצרכת לא ניתן להמיר במילות יחס אחרות וכל המוצרך אינו ניתן להמרה בתארי הפועל. המושא מצטרף לנשוא פעולי, כלומר, ניתן לומר: "אכל תפוח" ולא "יוסי תפוח". ישנם פעלים המצריכים אחריהם מושא אחד או יותר (למשל "יוסי הודה לחברו על המעשה").

בעברית קיימת בד"כ הבחנה בין שני סוגי מושאים: מושא ישיר ומושא עקיף.

---

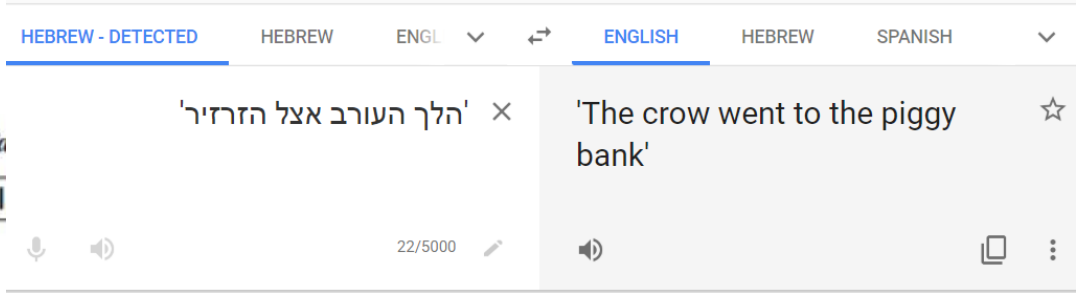
[3] הדוגמאות נלקחו מתוך וויקיפדיה העברית

- מושא ישיר - מושא ישיר הוא מושא המצטרף לנשוא בצורה ישירה, למשל: "יוסי אכל תפוח", כאשר המילה "תפוח" היא המושא.
- מושא עקיף - מושא עקיף הוא מושא המצטרף לנשוא באמצעות מילת יחס המוצרכת לפועל, כלומר מילת היחס היחידה שיכולה לבוא עם פועל זה, או כזו שמשפיעה על משמעותו. דוגמאות: "יוסי הודה לחברו", "יוסי הודה במעשה".

2.5 טיפולוגיה של סדר מילים

טיפולוגיה בבלשנות היא תחום העוסק בניסיון לסווג שפות על סמך התחביר, המבנה והפונקציונליות שלהן. ניתן לסווג שפות בכמה אופנים, ואחד המודלים הנפוצים ביותר הוא לסווג אותם הוא לפי סדר המילים הדומיננטי שלהם, המכונה גם טיפולוגית סדר מילים (Word Order Typology).

סדרי המילה העיקריים מתרחשים בסעיפים המורכבים מחלקי המשפט - נושא, נשוא ומושא. כמובן, לא כל הסעיפים משתמשים בנושא, נשוא ומושא, ולעתים קרובות הנושא והמושא אינם משמשים בשפות מסוימות. עם זאת, קיימות כמה מחלקות שונות שניתן להשתמש בהן לסיווג מגוון רחב של שפות. סקר שנערך על ידי ראסל טומלין בשנת 1986, שסיכום שלו **באיור 8**, מראה כמה תובנות הנגזרות מהניתוח שלו.



איור 9 גוגל תרגום דוגמא לסדר מילים

2			Spanish
3	Verb-Subject-Object	37 (9%)	Hebrew, Irish, Filipino, Aramaic
4	Verb-Object-Subject	12 (3%)	Baure, Malagasy, Aneityan
5	Object-Verb-Subject	5 (1%)	Apalai, Hixkaryana, Arecua
6	Object-Subject-Verb	1 (0%)	Warao

איור 8 טבלת סדר תפקידים תחבירית

באיור 8 אנו יכולים לראות שישנן שש מחלקות עיקריות של סידור מילים, ושפות כמו עברית עוקבות אחר מחלקות המילים פועל (נשוא) - נושא - מושא. דוגמא פשוטה יכולה להיות המשפט *'הלך העורב אצל הזרזיר'*, 'העורב' הוא הנושא, 'הלך' הוא הפועל, 'אצל

הזרזיר' הוא המושא. רוב השפות מהטבלה עוקבות אחר סדר המילים נושא-מושא-פועל. אפשר לראות זאת על ידי התרגום לאנגלית של אותו משפט **באיור 9.**

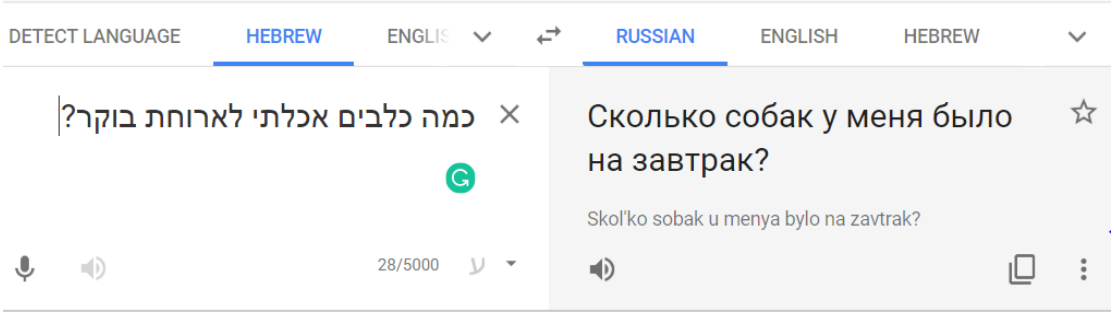
אפשר לראות כי הנושא הפך להיות ראשון במשפט באנגלית ואילו בעברית הנושא הוא השני במשפט. איור זה נותן לנו אינדיקציה לחשיבות סדר המילים וכיצד ייצוג המסרים יכול להיות שונה מבחינה דקדוקית בשפות שונות. וזה מביא אותנו לסיום הדיון שלנו בתחביר ובמבנה של שפות. בשלב הבא נבחן כמה מהמושגים סביב סמנטיקה של שפה.

2.6 סמנטיקה

ההגדרה הפשוטה ביותר של סמנטיקה היא לימוד המשמעות. לבלשנות יש תחום משלה של סמנטיקה לשונית, העוסק בחקר המשמעות בשפה, בקשרים בין מילים, ביטויים וסמלים, וציון משמעותם וייצוג הידע שהם מסמנים. במילים פשוטות, סמנטיקה עוסקת יותר בהבעות הפנים, הסימנים, הסמלים, שפת הגוף והידע המועברים כאשר מועברים מסרים מישות אחת לאחרת. ישנם ייצוגים שונים לתחביר וכללים זהים, כולל צורות דקדוק שונות שעסקנו בסעיפים הקודמים. ייצוג סמנטיקה באמצעות כללים או מוסכמות פורמליות היה תמיד אתגר בבלשנות. עם זאת, ישנן דרכים שונות לייצג משמעות וידע המתקבלים משפה.

3 עיבוד שפה טבעית (NLP)

הזכרנו את המונח "עיבוד שפות טבעיות" מספר רב של פעמים בפרק זה. נכון לעכשיו, אולי הצלחנו ליצור רעיון כלשהו לגבי המשמעות של NLP כעת נעבור לפירוט. NLP מוגדר כתחום במדעי המחשב והבינה המלאכותית המשלב בלשנות חישובית. תחום זה עוסק בעיקר בתכנון ובניית יישומים המאפשרים אינטראקציה בין מכונות לשפות טבעיות שהתפתחו על ידי בני האדם. שילוב זה הופך את ה- NLP לקשור למערכות של משימות המפתחות אינטראקציה בין בני האדם ומחשבים (Human Computer Interaction - HCI). טכניקות NLP מאפשרות למחשבים לעבד ולהבין שפה אנושית טבעית בכדי לספק פלט שימושי. בשלב הבא נדבר על כמה מהיישומים



איור 10 גוגל תרגום

העיקריים של NLP.

3.1 יישומיים עיקריים ב- NLP

תרגום מכונה – Machine Translation

תרגום ע"י מכונה הוא אולי אחד היישומים הנחשקים והמבוקשים ביותר בתחום ה- NLP. תחום זה מוגדר כטכניקה המסייעת במתן תרגום תחבירי, דקדוקי וסמנטי נכון בין שתי שפות. בראשית הדרך תחום זה היה הראשון במחקר ופיתוח ב- NLP. ברמה הפשוטה. תרגום מכונה הוא תרגום של שפה טבעית המתבצעת על ידי מכונה. כברירת מחדל, אבני הבניין הבסיסיים לתהליך תרגום המכונה כרוכים בהחלפה פשוטה של מילים משפה לשפה, אך במקרה כזה אנו מתעלמים מדברים רבים כמו דקדוק ועקביות במבנה נוסח. לפיכך, טכניקות מתוחכמות יותר התפתחו לאורך זמן, כולל שילוב של משאבים גדולים של תאגידי טקסט יחד עם טכניקות סטטיסטיות ושפות. אחת ממערכות התרגום הפופולריות ביותר היא Google Translate (**איור 10**). עם הזמן, מערכות תרגום מכונה הולכות ומשתפרות ומספקות תרגומים בזמן אמת תוך כדי כתיבה ודיבור.

זיהוי דיבור - Speech Recognition

מערכות זיהוי דיבור היא אולי היישום הקשה ביותר בין משימות NLP. לפי מספר חוקרים זהו מבחן האינטליגנציה הקשה ביותר במערכות בינה מלאכותית (מבחן טיורינג). עם הזמן חלה התקדמות רבה בתחום על ידי שימוש בטכניקות כמו סינתזת דיבור, ניתוח, ניתוח תחביר וסוגת הקשר. אך עדיין קיימת מגבלה ראשית אחת למערכות זיהוי דיבור : מערכות אלה הן מאוד ספציפיות לתחום מסוים ובמקרים רבים הן לא תעבודנה אם המשתמש ישנה אפילו מעט מהתוצאות הצפויות של התוכנה הנדרשות על ידי המערכת. מערכות זיהוי דיבור נמצאות כיום במקומות רבים, החל ממחשבים, טלפונים ניידים ועד מערכות סיוע וירטואליות. מערכות שאלה-תשובה (Question Answering System - QAS) מבוססות על העיקרון של מענה על שאלות, המבוסס על שימוש בטכניקות של NLP ושליפת מידע. הצלחה מסוימת בתחום זה הושגה בעזרת כלים כמו *SIRI*, אך היקפם עדיין מוגבל מכיוון שהם ממוקדים רק בתת-קבוצה של סעיפי מפתח וביטויים בשפה הטבעית.

הבנת הקשר של מילה Contextual Recognition and Resolution

תחום זה מכסה חלק גדול בהבנת השפה הטבעית וכולל גם שימוש תחבירי וסמנטי. פירוש של מילה מבחינת המילה עצמה הוא יישום פופולרי, בו אנו מנסים לגלות את המובן ההקשרי שלה במשפט נתון. קחו למשל את המילה ספר, מילה זו יכולה להתכוון לאובייקט המכיל ידע ומידע בטקסט כאשר הוא משמש כשם עצם, והיא יכולה גם להתכוון למקצוע כאשר משתמשים בו כתואר. איתור ההבדלים האלה במשפטים על בסיס הקשר הם הנחת היסוד העיקרית לפירוק משמעות המילים.

סיכום טקסטים - Text Summarization

המטרה העיקרית בסיכום טקסט היא לקחת קורפוס של מסמכי טקסט - שיכולים להיות אוסף של טקסטים, פסקאות או משפטים - ולהקטין את התוכן באופן המתאים ליצירת סיכום ששומר על נקודות המפתח של האוסף. סיכום ניתן לבצע על ידי התבוננות במסמכים השונים וניסיון לגלות את מילות המפתח, הביטויים והמשפטים שיש להם בולטות חשובה באוסף כולו. שני סוגים עיקריים של טכניקות לסיכום טקסט כוללים סיכום מבוסס חילוץ וסיכום מבוסס הפשטה. עם הופעת כמויות אדירות של טקסט ונתונים לא מובנים (unstructured), הצורך בסיכום טקסט בכדי להגיע לתובנות חשובות במהירות הוא ביקוש רב. מערכות סיכום טקסט בדרך כלל מבצעות שני סוגים עיקריים של פעולות. הראשון הוא סיכום גנרי, שמנסה לספק סיכום כללי של אוסף המסמכים הנבדקים. הסוג השני של הפעולה הוא סיכום מבוסס שאילתה, המספק סיכומי טקסט רלוונטיים לשאילתה שבהם הסינון של הקורפוס מסנן עוד יותר על סמך שאילתות ספציפיות, מילות מפתח וביטויים רלוונטיים מחולצים המתייחסים לשאילתה והסיכום נבנה.

סיווג טקסט - Text classification

המטרה העיקרית של סיווג טקסט היא לזהות לאיזו קטגוריה או כיתה יש להציב מסמך ספציפי על סמך תוכן המסמך. זהו אחד היישומים הפופולריים ביותר ב- NLP וענף למידת

המכונה מכיוון שעם הנתונים הנכונים קל מאוד להבין את העקרונות העומדים מאחורי הטקסט וליישם מערכות סיווג טקסט יעילות. ניתן להשתמש בטכניקות למידת מכונה supervised ו- unsupervised לפתרון בעיה זו, לעיתים משתמשים בשילוב של שתיהן. כלי זה עוזר לבנות הרבה יישומים מצליחים ומעשיים, כמו מסנני דואר זבל, סיווג כתבות חדשותיות ועוד.

3.2 ניתוח טקסטים - Text Analytics

כאמור, עם התפתחות הטכנולוגיה והופעת האיכות האדירה של כוח המחשוב והעיבוד, וכן עם הריבוי הגדול של נתונים לא מובנים (unstructured),לא עבר זמן רב וניתוח טקסטים החל לקבל תשומת לב רבה. עם זאת, ניתוח טקסטים מציב אתגרים מסוימים בהשוואה לשיטות ניתוח רגילות. טקסט הוא חומר שזורם ואינו מובנה (unstructured). רק לעיתים רחוקות אפשר לזהות דפוס ספציפי כלשהו - כמו נתוני מזג אוויר או פיצ'רים מובנים במאגרי מידע ריאלציונים. מכאן ששיטות סטטיסטיות סטנדרטיות אינן מועילות כאשר הן מיושמות על נתוני טקסט לא מובנים. פרק זה כיסה כמה מהמושגים העיקריים בניתוח טקסטים וכן דן בהגדרה וההיקף של ניתוח טקסטים כאשר המטרה הייתה לתת מושג רחב מה הרקע התיאורטי הקיים בתחום. ניתוח טקסטים, המכונה גם כריית טקסטים, הוא המתודולוגיה והתהליך שמבצעים כדי להפיק מידע ותובנות איכותיות וניתנות לפעולה מהנתונים הטקסטואליים. כל זה כרוך בשימוש ב- NLP, שליפת מידע וטכניקות למידת מכונה כדי לנתח נתוני טקסט לא מובנים לצורות מובנות יותר ולהפיק דפוסים ותובנות מנתונים אלה שיעזרו למשתמש הקצה. ניתוח טקסטים כולל אוסף של טכניקות למידת מכונה, לשון וסטטיסטיקה המשמשות למודל ולחילוץ של מידע מטקסט בעיקר לצורכי ניתוח, כולל ניתוח מודיעיני, חקר, תיאורי וניבוי. להלן כמה מהטכניקות והפעולות העיקריות בניתוח טקסט:

- סיווג טקסט
- אשכול טקסטים
- סיכום טקסט
- ניתוח משמעות
- זיהוי ישויות

ביצוע ניתוח טקסטים הוא לעיתים תהליך מעורב יותר מאשר ניתוח סטטיסטי רגיל או למידת מכונה. לפני שמיישמים טכניקה או אלגוריתם למידה כלשהו, יש צורך להמיר את נתוני הטקסט הלא מובנים לתבנית המקובלת על ידי אותם אלגוריתמים.

בהגדרה, גוף טקסט שנמצא תחת ניתוח הוא לרוב מסמך או משפטים, ועל ידי יישום של טכניקות שונות אנו בדרך כלל ממירים מסמך ומשפטים אלו לווקטורים של מילים. כל

ווקטור הוא מערך מספרי שערכיו הם משקלים ספציפיים לכל מילה שיכולה להיות התדר שלו, משמעותו, או תיאורים שונים אחרים - שחלקם נחקור בפרק 5.2. לעיתים קרובות יש צורך בשלב של ניקוי ועיבוד הטקסט כדי להסיר אלמנטים ונתונים רועשים.שלב זה נקרא עיבוד טקסט מראש (pre-processing). רק ברגע שיש לנו את הנתונים בתבנית הניתנת לקריאה ומובנת במכונה, נוכל להחיל אלגוריתמים רלוונטיים על סמך הבעיה שיש לפתור בהישג יד.

היישומים של ניתוח טקסטים הם רבים. הפופולאריים ביניהם כוללים את המשימות הבאות :

- איתור דואר זבל
- קטגוריית מאמרים חדשותיים
- ניתוח ומעקב מדיה חברתית
- ביו-רפואי
- מודיעין אבטחה
- ניתוח רגש
- מיקומי מודעות
- צ׳אט בוטים
- עוזרים וירטואליים (siri וכדו׳).

4 למידת מכונה – Machine Learning

למידת מכונה (לעיתים מכונה גם למידה חישובית) היא תת-תחום במדעי המחשב ובבינה מלאכותית המשיק לתחומי הסטטיסטיקה והאופטימיזציה. התחום עוסק בפיתוח אלגוריתמים שמטרתם לאפשר למחשב ללמוד מתוך הנתונים, ומתוך למידה זו לפעול במגוון משימות חישוביות שעד היום לא הצלחנו לבצע בתכנות הקלאסי. שני תחומים מקבילים ללמידת מכונה הם תחום כריית מידע (כריית נתונים) ותחום זיהוי תבניות (זיהוי תבניות) שרבים מן הכלים והאלגוריתמים שפותחו בו משותפים לתחומים האלה.

נעשה חזרה קצרה - נהוג לחלק את אלגוריתמי למידת המכונה למספר סוגים:

למידה מונחית (supervised learning) – בסוג זה של למידה כל דוגמה מגיעה יחד עם תווית סיווג. מטרת האלגוריתם היא לחזות את הסיווג של דוגמאות חדשות שאותן לא פגש בתהליך הלמידה. אימון של רשת עצבית מלאכותית ("**רשת נוירונים**") מסתמך על אלגוריתמים מסוג זה.

למידה בלתי מונחית (unsupervised learning) – בסוג זה של למידה מטרת האלגוריתמים היא למצוא ייצוג פשוט וקל להבנה של אוסף הנתונים. שיטות נפוצות מסוג זה הן חלוקה לצברים (clustering), והטלה ליריעות ממד נמוך כגון ניתוח גורמים ראשיים (PCA).

למידת חיזוק (reinforcement learning) – בסוג זה של למידה אלגוריתם הלמידה מקבל משוב חלקי על ביצועיו (רק לאחר סיום ביצוע המטלה) ועליו להסיק אילו מהחלטותיו הביאו להצלחה/כישלון.

במחקר זה כאמור נתמקד בסגנון הלמידה המונחית כאשר תהליך הלמידה יתבצע בעזרת רשת נוירונים.

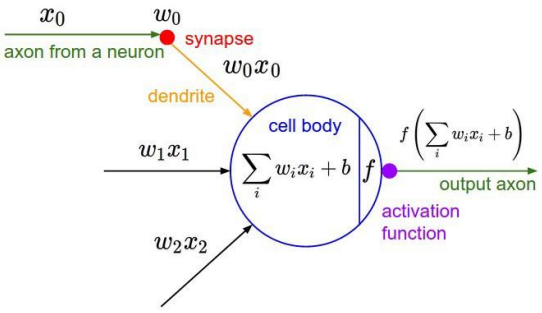
4.1 רשתות נוירונים - neural network

4.1.1 הגדרה פורמאלית

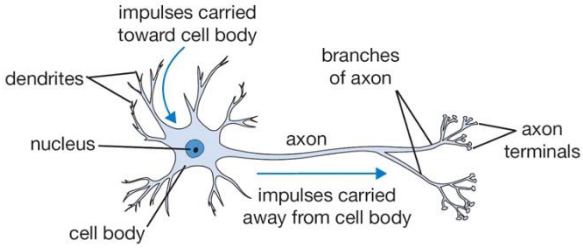
ההגדרה של רשת נוירונים, המכונה יותר נכון רשת עצבית 'מלאכותית' (Artificial Neuarl Network -ANN), ניתנה על ידי הממציא של אחד ממחשבי הנוירונים הראשונים, ד"ר רוברט הכט-נילסן. הוא הגדיר רשת נוירונים כ: "... *מערכת מחשוב המורכבת ממספר מרכיבי עיבוד פשוטים ומחוברים זה לזה, המעבדים מידע על ידי תגובת המצב הדינאמי שלהם לתשומות חיצוניות.*"[4]

ניתן לחשוב גם על רשת נוירונים מלאכותית כמודל חישובי אשר נוצר בהשראת האופן בו רשתות עצביות ביולוגיות במוח האנושי מעבדות מידע.

[4] "a computing system made up of a number of simple, highly interconnected processing elements, ... which process information by their dynamic state response to external inputs."



איור 12 נוירון חישובי



איור 11 נוירון ביולוגי

4.1.2 מוטיבציה ביולוגית וקשרים

היחידה החישובית הבסיסית של המוח היא נוירון. במערכת העצבים האנושית ניתן למצוא כ- 86 מיליארד נוירונים, והם קשורים לסינפסות של 10 - 10^{15}. האיורים שלהלן מראים ציור מצויר של נוירון ביולוגי (**איור 11**) ומודל מתמטי נפוץ (**איור 12**). יחידת החישוב הבסיסית ברשת עצבית היא העצב, המכונה לעתים קרובות צומת או יחידה. יחידה זו מקבלת קלט מכמה צמתים אחרים, או ממקור חיצוני ומחשבת כך את הפלט. לכל קלט משויך משקל (w) המוקצה על בסיס חשיבותו היחסית לקלטים האחרים. באופן כללי הצומת מחשב פונקציה על הסכום המשוקלל של קלטיו כמתואר באיור 12. הרעיון הוא שהעוצמות הסינפטיות (המשקולות w) ניתנות ללמידה והן בעצם אלה ששולטות על מידת ההשפעה והכיוון שלה באמצעות שתי פעולות בסיסיות: עירור - excitory (משקל חיובי) או מעכב - inhibitory (משקל שלילי) של נוירון אחד אחר. במודל הבסיסי, הדנדריטים (dendrites) נושאים את האות לגוף התא, שם הם מחושבים לסכום. אם הסכום הסופי הוא מעל סף מסוים, הנוירון יכול לירות ולשלוח ספייק (spike) לאורך האקסון שלו. במודל החישובי אנו מניחים כי לתזמון המדויק של הספייקים אין חשיבות, וכי רק תדירות הירי היא שמעבירה מידע. אנו מדגמים את קצב הירי של הנוירון בעזרת פונקציית הפעלה (Activation Function) המייצגת את תדירות הספייקים לאורך האקסון.

4.1.3 ארכיטקטורת רשת נוירונים

מההסבר לעיל ניתן להסיק כי רשת עצבית עשויה מנוירונים. באופן הביולוגי הנוירונים מחוברים דרך סינפסות בהן זורם מידע (שהן המקבילות למשקולות במודל החישובי), כאשר אנו מאמנים רשת נוירונים אנו רוצים שהנוירונים יעבירו את המידע בכל פעם שהם לומדים משקלות ספציפיים מדפוסים מהנתונים, ואנחנו מנרמלים את קצב התעבורה באמצעות פונקציית הפעלה. המבנה של הרשת מורכב ממספר סוגי צמתים/שכבות:

<u>שכבת קלט (**input layer**)</u>: בצמתים אלה לא נעשה חישוב בתוך השכבה. שכבות אלו רק מעבירות את המידע לשכבה הבאה (רובד מוסתר רוב הזמן). גוש צמתים נקרא גם שכבה.

שכבה חבויה (**hidden layer**): שכבות נסתרות הן המקום בו מתבצע עיבוד או חישוב ביניים. הן מבצעות חישובים ואז מעבירים את המשקולות (אותות או מידע) משכבת הקלט לשכבה הבאה (שכבה נסתרת אחרת או לשכבת הפלט) .

שכבת פלט (**output layer**): כאן אנו משתמשים סוף סוף בפונקציית הפעלה הממפה לפורמט הפלט הרצוי (למשל *SoftMax* לסיווג).

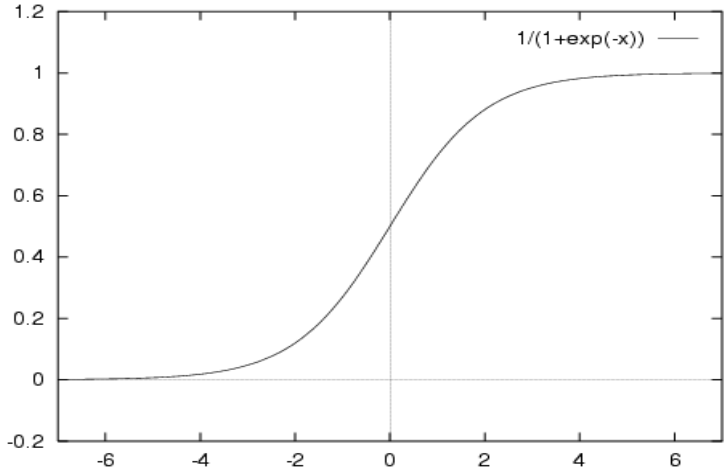
חיבורים ומשקולות: הרשת מורכבת מחיבורים, כאשר כל חיבור מעביר את הפלט של נוירון i לכניסה של נוירון j. במובן זה i קודמו של j ו- j הוא ממשיך דרכו של i, לכל קשר נקבע משקל W_{ij}.

4.1.4 פונקציית הפעלה - Activation function

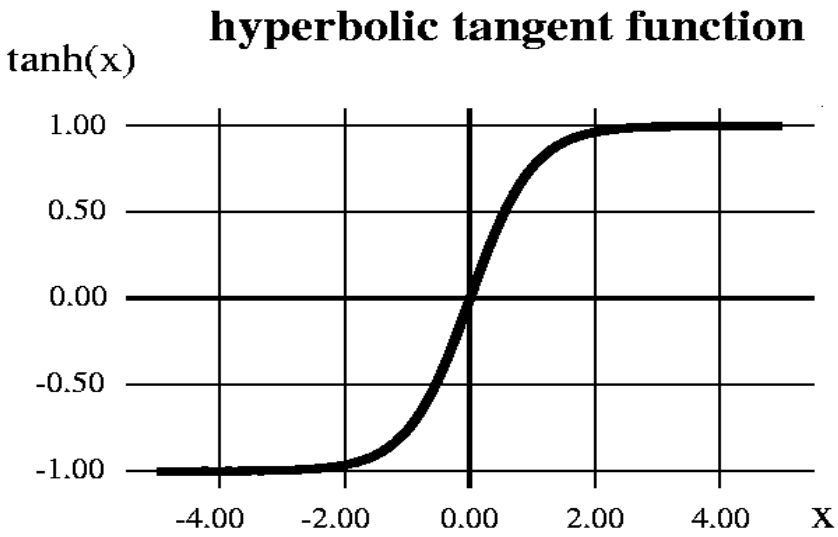
פונקציית ההפעלה של צומת מגדירה את הפלט של אותו צומת בהינתן קלט או קבוצת כניסות. ניתן לראות במעגל שבבי מחשב רגיל רשת דיגיטלית של פונקציות הפעלה שיכולות להיות "ON" (1) או "OFF" (0), בהתאם לכניסה. התנהגות זו דומה להתנהגותו של הפרספטרון (perceptron) הליניארי ברשתות עצביות. עם זאת, פונקציית ההפעלה הלא-לינארית מאפשרת לרשתות כאלה לחשב בעיות לא טריוויאליות באמצעות מספר קטן של צמתים. כלומר ברשת נוירונים פשוטה אנו סוכמים את סך הקלטים (X) והמשקלים התואמים שלהם (W) ומפעילים עליו פונקציית הפעלה $f(x)$ בכדי לקבל את הפלט של אותה שכבה ולהזין אותה כקלט לשלב הבא שכבה. ברשתות נוירונים פונקציה זו נקראת גם פונקציית ההעברה (function transfer).

ישנן מספר סיבות לשימוש בפונקציות אקטיבציה. להלן העיקריות שבהן:

- אם לא נשתמש בפונקציית האקטיבציה נקבל פשוט פונקציה לינארית ואז נהיה די מוגבלים בעבודה עם קלטים בעלי תבנית לא לינארית. כלומר רשת נוירונים לינארית שוות ערך למודל רגרסיה לינארית שכוחו מוגבל כידוע ולא מציג תוצאות טובות בחלק גדול הזמן.
- לא תמיד הקלט שלנו הוא לינארי. לדוגמא במקרים של קלטי וידאו תמונה או אודיו, אנו משתמשים בטכניקות של רשתות נוירונים כדי להבין משהו במערכות נתונים מורכבות, בעלי ממדיות גבוהה, שאינן ליניאריות מוחלטות או בכלל. במקרים אלה הדגם כולל המון שכבות נסתרות בין לבין ויש בו ארכיטקטורה מסובכת מאוד המסייעת לנו לנצל את ידע ממערכי נתונים גדולים ומורכבים כאלה.
- תכונה חשובה נוספת של פונקציית אקטיבציה – היותה ניתנת לגזירה. הגזירות היא הבסיס לביצוע אסטרטגיית אופטימיזציה של backpropagation תוך ביצוע propagating backwards ברשת כדי לחשב את הגרדיאנט (gradients) של שגיאה (Error (loss) ביחס למשקולות ובכך לייעל את המשקלות באמצעות Gradient descend או כל טכניקת אופטימיזציה אחרת כדי להפחית שגיאה.



איור 13 פונקציית סיגמואיד



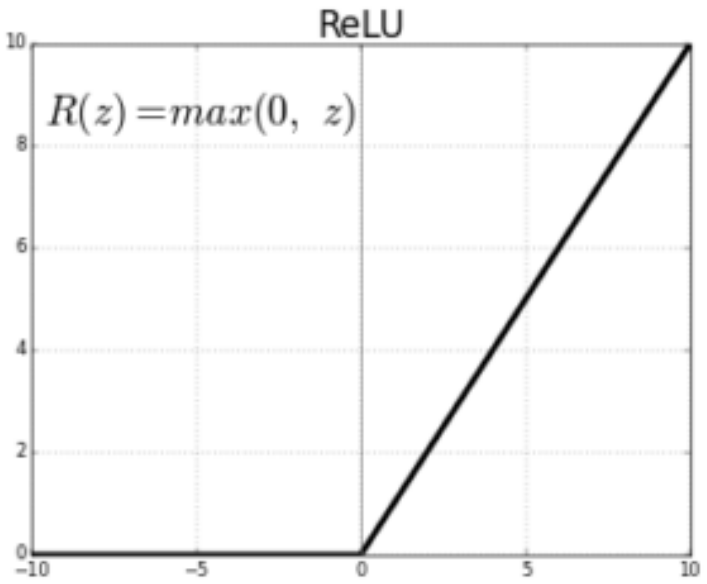
איור 14 פונקציית tanh

קיימים מספר רב של פונקציות הפעלה, נתייחס לעיקריות שבהן עם דגש על כאלו בהן השתמשנו בפרוייקט.

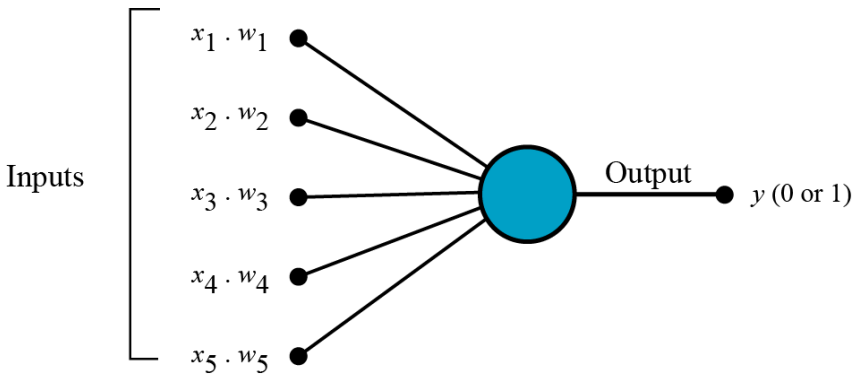
4.1.4.1 פונקציית הסיגמואיד - *Sigmoid*

איור 13: $f(x) = \frac{1}{1+e^{-x}}$. הטווח שלה הוא בין 0 ל -1. פונקציה זו פשוטה למימוש.

אחד החסרונות שלה הוא שהאפס אינו במרכז הפונקציה.



איור 15 פונקצית relu



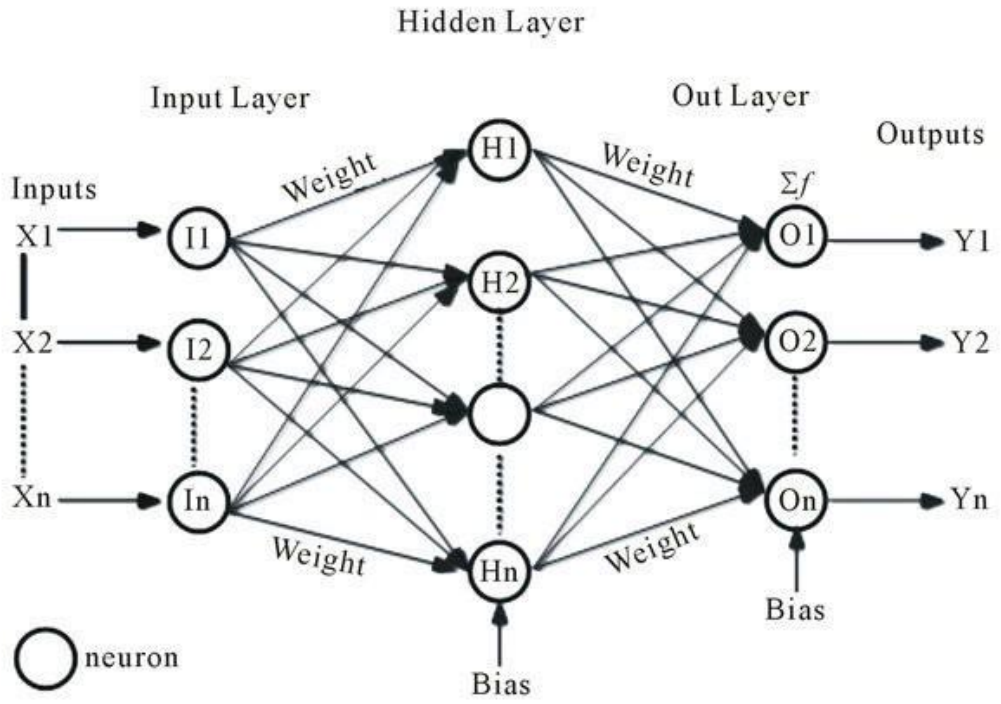
איור 16 נוירון חישובי בודד

4.1.4.2 *Hyperbolic Tangent function - Tanh*

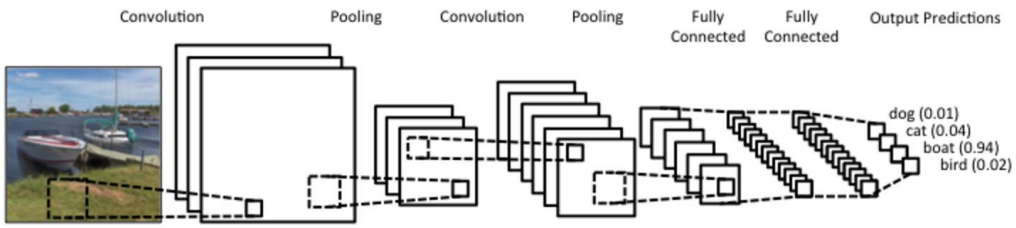
איור 14: $f(x) = \frac{1-e^{-2x}}{1+e^{-2x}}$ הפעם הפלט ממורכז באפס. תהליך האופטימיזציה בפונקציה זו קל יותר. מסיבה זו נעדיף אותה על פני סיגמואיד בחלק מהמקרים בשלבי האימון.

4.1.4.3 *ReLu - Rectified Linear units*

איור 15: $R(z) = \max(0, z)$ פונקציה זו הפכה למאוד פופולארית לאחרונה מכיוון שהוכח כי היא מציגה שיפור של פי שש במדד ההתכנסות מאשר tanh. ניתן לראות שהפונקציה הזאת פשוטה למימוש ויעילה מאוד. שימו לב, בלמידת מכונה אנו שמים דגש על הפשטות ברכיבים והעקביות.



איור 17 פרספטרון רב שכבתי



איור 18 CNN

ישנם סוגים רבים של רשתות עצביות ובתוך סוגים אלה יש גם סוגי משנה. נפרט בקצרה את הרשתות השימושיות ביותר.

4.1.5 Feedforward Neural Network

1. רשת נוירונים מסוג Feedforward היא רשת בה החיבורים בין היחידות אינם מתנהגות בצורה מחזורית. ברשת זו המידע נע בכיוון אחד בלבד, קדימה, מצמתי הקלט, דרך הצמתים הנסתרים (אם קיימים) עד צמתי הפלט. אין חזרות או לולאות ברשת. אנו יכולים להבחין בשלושה סוגים של רשתות נוירונים Feedforward:
   i. **פרספטרון (Perceptron) חד שכבתי- איור 16** זוהי הרשת העצבית הפשוטה ביותר העדכנית שאיננה מכילה שכבה נסתרת כלשהי, כלומר היא מורכבת רק משכבה אחת של צמתי פלט. קוראים לזה "חד שכבתי" מכיוון שאם אנו סופרים את השכבות איננו כוללים את שכבת הקלט. הסיבה לכך היא מכיוון שבשכבת הקלט לא מתבצעים חישובים, הכניסות מוזנות ישירות לפלטים דרך סדרת משקולות.

ii. **<u>פרספטרון (Perceptron) רב שכבתי (MLP)- איור 17</u>** - סוג רשת זה מורכב מכמה שכבות של יחידות חישוב, המחוברות בדרך כלל זו לזו קדימה. לכל נוירון בשכבה אחת יש קשרים לנוירונים של השכבה שלאחר מכן. ביישומים רבים היחידות של רשתות אלה מיישמות בפונקציית sigmoid כפונקציית אקטיבציה. MLP שימושיים מאוד מכיוון שהם מסוגלים ללמוד ייצוגים לא לינאריים (ברוב המקרים הנתונים שיוצגו בפנינו אינם ניתנים להפרדה לינארית).

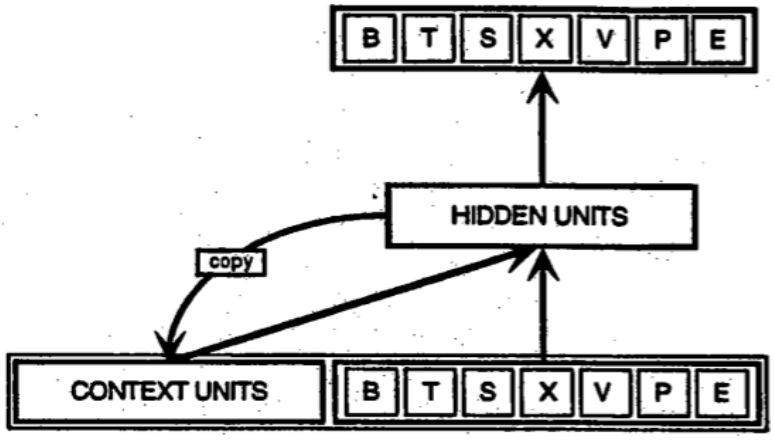
iii. **<u>Convolutional Neural Networks - CNN איור 18</u>** - רשתות נוירונים קונבולוציונליות דומות מאוד לרשתות נוירונים רגילות. הן מורכבות מנוירונים בעלי משקולות והטיות (biases) הניתנות ללמידה. ברשת נוירונים קונבולוציונית (CNN, או ConvNet), דפוס הקשרים בין הנוירונים נוצר בהשראת החלק במוח הנקרא קליפת הראייה[5], יחידות נוירונים מגיבות לגירויים רק באזור מוגבל של הרשת המכונה השדה הקולט. הכוונה היא לייצר שדות קליטה חופפים חלקית, כך שהן תכסינה את כל אופציות הקלט. היישומים הרחבים של רשתות מהסוג הזה קיים בעיקר בזיהוי תמונות ווידאו. בנוסף, חשוב לציין כי CNN דורש נתונים גדולים יותר להתאמן עליהם כתוצאה מהארכיטקטורה הייחודית שלו.

ברשתות feedforward תהליך ה-backpropagation עובר אחורה מהשגיאה הסופית (final error) דרך הקלטים, המשקולות והכניסות של כל שכבה נסתרת (hidden layer), על ידי חישוב הנגזרות החלקיות שלהן $\frac{dE}{dw}$, או הקשר בין אחוזי השינוי שלהן. נגזרות אלה משמשות במקרים אלה על ידי כלל הלמידה שלנו, gradient descent כדי לכוון את המשקולות כלפי מעלה או מטה, לכיוון שיגרום להפחתת השגיאה.

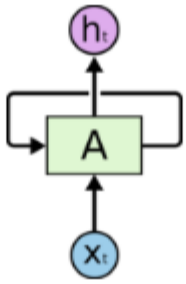
4.1.6 Recurrent neural networks (RNN)

רשתות חוזרות - Recurrent הן סוג של רשת נוירונים המיועדות לזהות דפוסים ברצפי נתונים, כמו טקסט, גנים, כתב יד, דיבור או בקצרה נתונים סדרתיים. היכולת להתעסק עם רצפים נובעת מכך שאלגוריתמים אלה לוקחים בחשבון זמן ורצף, כלומר יש להם מימד שמתייחס לזמן. מחקרים מראים כי רשתות חוזרות הן מהסוג החזק והשימושי ביותר של רשת נוירונים, כאשר הן באות לצד מנגנון Attention ורשתות Memory. יש לציין כי RNN ישים אפילו לתמונות הניתנות לפירוק לסדרת קטעים וכך לטפל בהן כרצף. מכיוון ש-RNN מחזיקות בסוג מסוים של זיכרון אנו נבצע אנלוגיות רבות לזיכרון במוח האנושי.

[5] קליפת הראייה (אנגלית: Visual cortex, נקראת גם קליפת המוח הראייתית) היא האזור המרכזי במוח בו מעובד מידע חזותי הנקלט במערכת הראייה. קליפת הראייה מהווה חלק מקליפת המוח, והמוקד שלה הוא באונה העורפית. המידע החזותי שנקלט ברשתית העין מועבר לקליפת הראייה דרך גרעין הברך הצדי שבתלמוס.



איור 19 יחידת RNN בודדת



Recurrent Neural Networks have loops.

איור 20 מבנה יחידי של נוירון ברשת חוזרת

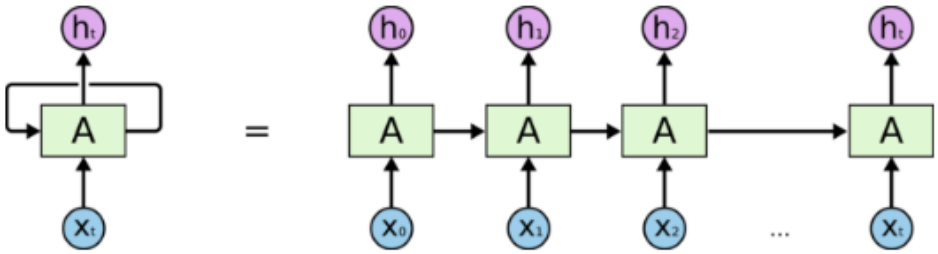
לעומת רשתות מסוג Feedforward, RNN מתייחסות לא רק לקלט הנוכחי שהן מקבלות, אלא גם למה שהן קיבלו בעבר. כפי שניתן לראות **באיור 19** של RNN פשוטה שם ה-***BTSXPE*** בתחתית השרטוט מייצג את דוגמת הקלט ברגע הנוכחי, ויחידת CONTEXT מייצגת את התפוקה של הרגע הקודם.

התוצאה שה-RNN הגיעה אליה בשלב $t-1$ משפיעה על התוצאה שתגיע ברגע אחד אחר כך בזמן t. מכאן אפשר להבין של-RNN יש שני מקורות קלט, **הקלט הנוכחי והקלט מהמצב הקודם**, כאשר העיקרון בכל הרשת הוא שילוב של שני הקלטים האלה כדי לקבוע כיצד הם מגיבים לנתונים חדשים.

אם כך, ההבדל העיקרי של ה-RNN מרשתות feedforward הוא אותה לולאת משוב המחוברת ליחידות העבר שלהן, תוך כניסת קלט באופן רציף. ניתן לומר שזאת בעצם הסיבה של-RNN יש מעין 'זיכרון' כיוון שהרשת מתייחסת לקלטים מהעבר. הוספת יכולת הזיכרון הזו לרשתות נוירונים הופכת את הרשת ליעילה יותר עבור מקרים בהם יש מידע המסתתר ברצף עצמו, כלומר יש חשיבות לסדר בו המידע מגיע.

אפשר להסביר זאת גם בדרך אחרת - מכיוון שלולאת המשוב הזו מתרחשת בכל שלב בסדרה של הקלטים, כל מצב נסתר מכיל עקבות מידע לא רק של המצב הנסתר הקודם, אלא גם של כל אלה שקדמו ל- h_{t-1} (כמובן כל עוד הרצף ממשיך). **באיור 20** ניתן לראות יחידה בודדת של RNN הקלט- x_t והפלט- h_t. הלולאה פנימה מאפשר למידע לעבור מיחידה אחת של הרשת אל היחידה הבאה. בעצם אפשר לחשוב על RNN כהרבה עותקים

של אותה רשת. כל יחידה מעבירה את המידע ליחידה הבאה. ניתן להסתכל על הלולאה בצורה פשוטה יותר כמו שמופיע **באיור 21**. פריסה זו של הרשת חושפת בפנינו את היתרון העיקרי שיש ל-RNN כאשר משתמשים בהם עבור נתונים סדרתיים כמו טקסטים גלי קול וכדומה.



An unrolled recurrent neural network.

איור 21 פריסה של נירון ברשת חוזרת

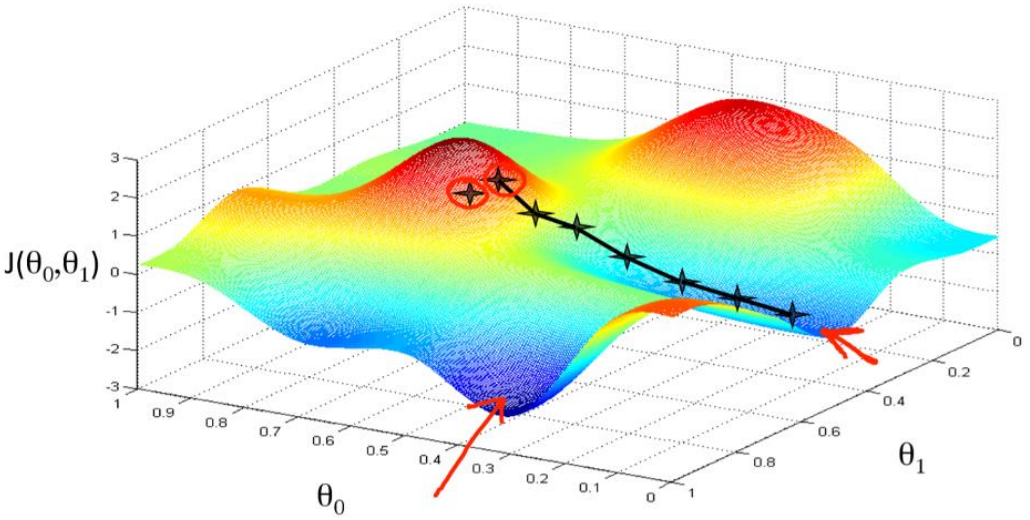
ניתן לתאר את התהליך הזה שמתרחש בRNN באמצעות נוסחה מתמטית פשוטה:

$$\mathbf{h}_t = \phi\left(W\mathbf{x}_t + U\mathbf{h}_{t-1}\right),$$

המצב הנסתר (hidden state) בשלב הזמן- t הוא h_t. הפונקציה מקבלת את ה**קלט** - x_t, מעדכנים את הקלט על ידי **מטריצת משקל** W שנוספה למצב הנסתר (hidden state) של השלב הקודם h_{t-1} כפול **מטריצת מצב נסתר למצב נסתר** (hidden-state-to-hidden-state matrix) U, הקרויה גם מטריצת מעבר (בדומה לשרשרת מרקוב[6]). מטריצות המשקל משמשות כעין פילטרים הקובעים מהי רמת החשיבות שיש להעניק הן לקלט הנוכחי והן

[6] כדוגמה, ניתן לתאר דגם פשטני של מזג האוויר כשרשרת מרקוב. בניית הדגם יוצאת מהפשטה על פיה מזג האוויר מתואר במשך שבוע כסדרה של 7 מצבים (מצב לכל יום): בהיר, מעונן או גשום. על מנת להפוך תיאור זה למודל הסתברותי, יש להגדיר התפלגות במרחב הסדרות, כלומר להתאים הסתברות להתרחשותה של כל סדרה של 7 מצבים. מודל זה, למשל יכול להתאים לסדרה: "גשום, גשום, מעונן, בהיר, בהיר, בהיר, בהיר" הסתברות 0.01, ולסדרה הקבועה "גשום, גשום, ... גשום" הסתברות 0.2, וכך הלאה. (ויקיפדיה העברית)

למצב הנסתר בעבר. ה**שגיאה** שהם מייצרים תתוקן באמצעות backpropagation ותשמש להתאמת משקולותיהם עד הגעה למינימום.



איור 22 תהליך הback-propagation

הפלט והמצב הנסתר (hidden state) מחושבים על ידי הפונקציה φ - פונקצית sigmoid או tanh, המהוות כלי סטנדרטי לעיבוי ערכים גדולים או קטנים מאוד לתוך טווח קבוע, כמו גם חישוב הגרדיאנט עבור תהליך ה-backpropagation .

בהינתן סדרה של אותיות, RNN תשתמש בתו הראשון כדי לנסות לחזות את הערך של התו השני, כך שהאות א׳ בהתחלה תוביל אותה סתם לדוגמא להסיק שהאות הבאה תהיה ב׳, ואילו ט׳ עשויה להוביל אותה להסיק שהאות הבאה תהיה מ׳.

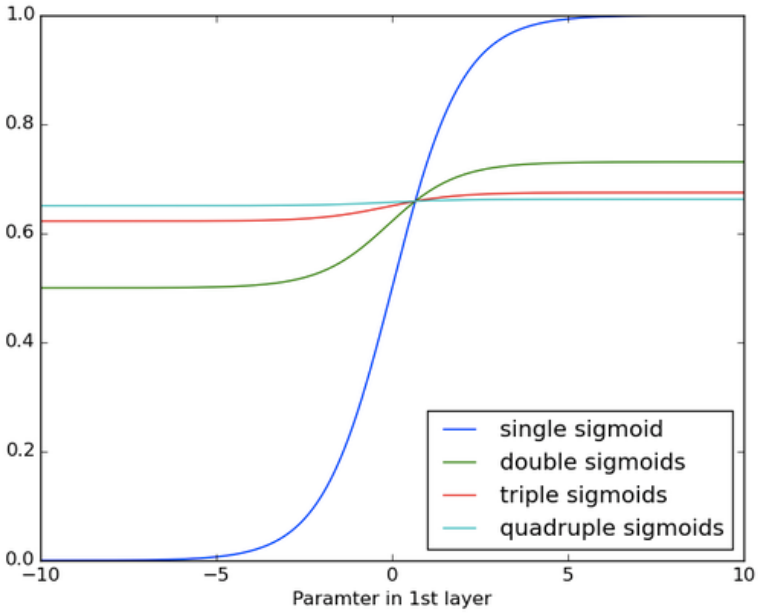
נזכיר, מטרתן של RNN היא לסווג במדויק את הקלט הרציף. כמובן שאנו מסתמכים על תהליך backpropagation של הטעות (error) וירידה ב-gradient כדי לעשות זאת, רק שבמקרים סדרתיים RNN יבצע זאת בצורה טובה יותר. חשוב להזכיר כי בRNN אנו משתמשים בהרחבה של תהליך ה- backpropagation הנקרא *פעפוע לאחור לאורך זמן׳* (backpropagation through time), או בקיצור BPTT. הוספת הזמן לתהליך, במקרה זה, מתבטא בפשטות בסדרת חישובים מוגדרת היטב, המקשרת בין השלבים צעד אחר צעד, שאת כל זה תהליך ה- backpropagation צריך לחשב.

אולם למרות כל היתרונות שצוינו גם RNN לא נקי מבעיות. הבעיה העיקרית היא בעיית הVanishing (and Exploding) Gradients .

4.1.6.1 Vanishing (and Exploding) Gradients

כמו רוב רשתות הנוירונים, RNN לא הומצאו לאחרונה. כבר בראשית שנות התשעים, בעיית ה-vanishing gradient התגלתה כמכשול המרכזי לביצועים של ה-RNN. כשם שקו ישר מבטא שינוי ב- x לצד שינוי ב- y, gradient מבטא את השינוי בכל המ**שקולות** ביחס לשינוי ב**טעות**. אם איננו יכולים לדעת את ה gradient, איננו יכולים להתאים את המשקולות לכיוון כך שכיוון זה יוריד את השגיאה (**איור 22**), והרשת שלנו תפסיק ללמוד. RNN מבקשות ליצור קשרים בין

פלט סופי לבין תוצרים מוקדמים יותר בתהליך החיזוי. רצון זה נתקל בקשיים רבים מכיוון שקשה מאוד לדעת כמה חשיבות להעניק לקלטים שמרוחקים מאוד מהתוצאה הרצויה. (ניתן לחשוב על זה כירייה של תותח למרחק רב, עקב המרחק הרב קשה מאוד להעריך לאיזה כיוון מדויק בהתאם לרוח ומשתנים אחרים יש לירות את הפגז כדי להגיע למטרה הסופית). ההסבר המתמטי לקושי הזה נובע בחלקו מהסיבה שהמידע הזורם דרך רשתות נוירונים ארוכות עובר בשלבים מרובים של כפל עד למצב שבו הכפל שהתבצע בשלבים הראשונים איבד את ההשפעה שלו על הפלט הסופי. התוצאה הסופית היא שנגזרות רגישות עלולות להיעלם או 'להתפוצץ' מה שגורם לתהליך ה gradient descent לאבד מערכו.



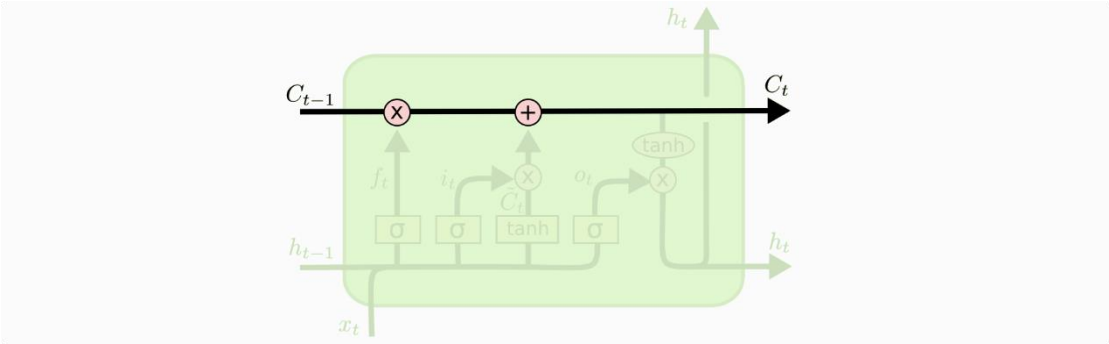
איור 23 השפעה של הפעלת פונקציית הסיגמואיד מס' פעמים

בעיית Exploding gradients מתייחסת לכל משקל ברשת כמו באפקט הפרפר הידוע שכנפיו המתנופפות גורמות להוריקן מרוחק. כלומר השיפועים של המשקלים הללו גדלים בקצב מטורף כך שהערכים מאבדים את היחס לתוצאה הרצויה. אך ניתן לפתור את הבעיה בקלות יחסית, מכיוון שניתן לחלק אותם או לצמצם. בעיית ה-Vanishing Gradients מתייחסת למצב בו המשקלים כ"כ קטנים עד שהם כבר מאבדים מערכם והרשת לא יכולה להמשיך בתהליך הלמידה. לבעיה זו הפתרון קצת יותר מורכב וקשה. ב**איור 23** ניתן לראות את ההשפעה של הפעלת פונקציית sigmoid שוב ושוב. הנתונים הופכים ל'שטוחים' יותר ויותר עד

שבמתיחות גדולות השיפוע שלהם קשה לזיהוי. תהליך זה המוצג באיור מקביל לתהליך בו השיפוע נעלם כאשר הוא עובר בשכבות רבות בתוך הרשת העמוקה.

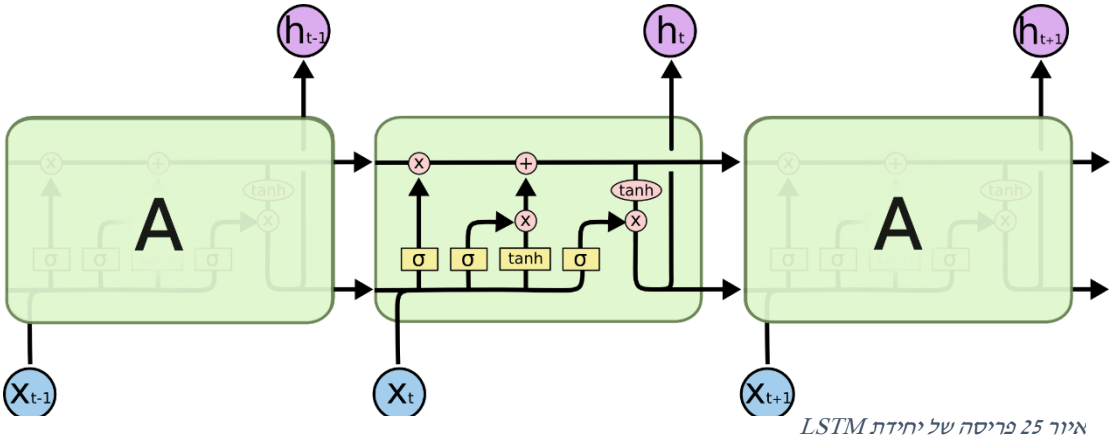
4.1.6.2 Long Short-Term Memory

באמצע שנות ה -90 הציעו קבוצת חוקרים מגרמניה בראשות ספ-הוכרייטר ויורן שמידאובר[7] וריאציה של RNN בתוספת מה שהם כינו כ- ״יחידות זיכרון לטווח קצר - ארוך״, Long Short-Term Memory או בקיצור LSTM. לפי המחקר שלהם יחידות אלו פותרות את בעיית ה-vanishing gradient על ידי שמירה של ערך השגיאה לאורך זמן ולאורך שכבות. בעקבות השמירה על ערך השגיאה בצורה קבועה יותר, ניתן לגרום ל-RNN להמשיך בתהליך הלמידה לאורך שלבי זמן רבים, ובכך לפתח אפשרויות של קישור בין גורמים ותופעות המתרחשות במרחק רב אחת מהשנייה. יש לציין כי זהו אחד האתגרים המרכזיים של למידת מכונה ו- AI, מכיוון שאלגוריתמים של בינה מלאכותית מתמודדים לעתים קרובות עם סביבות בהן אירועים בעלי ערך חשוב באים לידי ביטוי במציאות באופן דליל או מאוד מאוחר, כמו בחיים עצמם. ברשתות LSTM הארכיטקטורה מאפשרת העברת מידע מחוץ לזרם הרגיל של ה-RNN. ברשתות מסוג LSTM ניתן לשמור מידע, לכתוב או לקרוא מתוך התא, בדומה לשמירת נתונים בזיכרון של מחשב. התא מקבל החלטות לגבי מה לאחסן, ומתי לאפשר קריאה, כתיבה ומחיקה, דרך שערים פנימיים שנפתחים ונסגרים בהתאם ללמידה. בניגוד לאחסון הדיגיטלי במחשבים, שערים אלה הם אנלוגיים, המיושמים באמצעות כפל בצורה על ידי פונקציית sigmoid, אשר נותן תוצאות

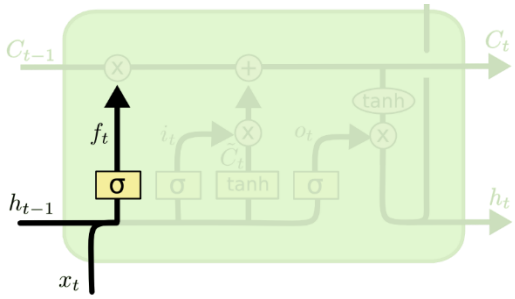


איור 24 יחידת LSTM

[7] Sepp Hochreiter and Juergen Schmidhuber



איור 25 פריסה של יחידת LSTM

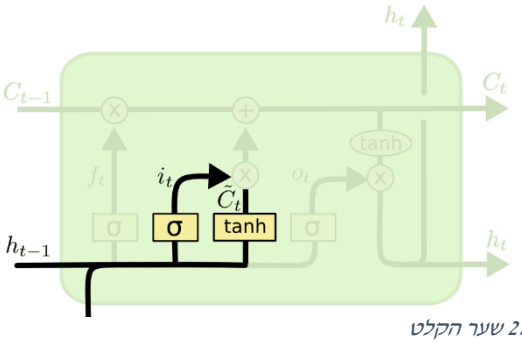


איור 26 שער השכחה בLSTM

בטווח של 0-1. לעבודה בצורה אנלוגית יש יתרון על פני הדיגיטלית מכיוון שהיא דיפרנציאלית, ולכן היא מתאימה יותר לתהליך ה-backpropagation שמתבצע בעיקר בצורה זו. השערים שביחידות ה-LSTM פועלים לפי האותות שהם מקבלים, ובדומה לצמתים קלאסיים של רשת נוירונים, הם חוסמים או מעבירים מידע על סמך חוזקו ודרך הייבוא שלו. את האותות הם מסננים באמצעות מערכות המשקלים שלהם. המשקלים הללו, כמו המשקולות שמווסתים קלט ומצבים נסתרים (hidden states), מותאמים באמצעות תהליך הלמידה של ה-RNN. כלומר, התאים לומדים מתי לאפשר לנתונים להיכנס, לעזוב או להימחק בתהליך האיטראטיבי של ניחושים, backpropagating error, שינוי המשקולות ו-gradient descent. **באיור 24** ניתן לראות את יחידת ה LSTM – כחתך מהרשת הכללית. ראשית, המפתח לכל הLSTM, הוא ה-**cell state** באיור הוא בא לידי ביטוי בקו האופקי שעובר בין היחידות בצורה איטרטיבית, כפי שמופיע ב**איור 25** בחלק העליון של הדיאגרמה. ה- cell state הוא סוג של מסוע בין היחידות והוא עובר ישר בין כל השרשרת של הרשת, כאשר יש מס׳ קטן של פעולות לינאריות בתוך המעבר הזה. ל-LSTM יש את היכולת להסיר או להוסיף מידע ל- cell state אשר הערך המספרי שלו מסומן באיור כ- C_t (זה בעצם סוג של הפלט). יכולות אלה מתאפשרות בזכות שערים שנסביר עליהם בהמשך. השערים מורכבים משכבה של פונקציית אקטיבציה מסוג סיגמואיד, ופעולה של כפל[8]. שכבת הסיגמואיד כידוע מוציאה מספרים בין אחת לאפס, ומתארת כמה מידע מכל רכיב צריך לעבור הלאה. ערך של אפס פירושו ״שום דבר לא עובר״ ואילו ערך של אחד פירושו ״הכול עובר״. לLSTM יש שלושה שערים מהסגנון הזה. כעת נסביר שלב אחרי שלב את הרכיבים שמרכיבים יחידת LSTM. יש לציין כי הLSTM

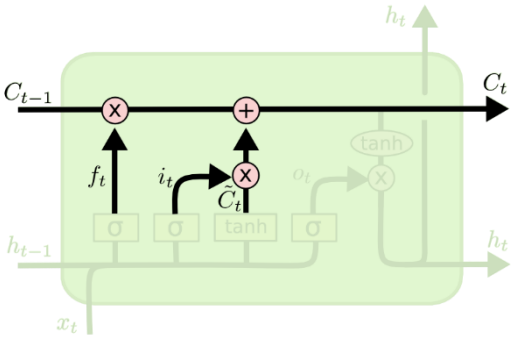
[8] pointwise

שהשתמשנו בו במחקר זה מתאים **לגרסה המקורית**[9] שלו. למרות שיש לו עוד הרחבות העדפנו ללכת על המקור.



איור 27 שער הקלט

בשלב הראשון נחליט איזה מידע יעבור מה-cell state. החלטה זו תתבצע בעזרת השער



איור 28 שער העדכון

הראשון שלנו והוא – "שער השכחה" (forget gate) כמתואר ב**איור 26**. שער השכחה מקבל שני ארגומנטים, h_{t-1} שהוא המצב הנסתר (hidden state) הקודם. והשני הוא הקלט x_t. הפלט של השער הזה יהיה מספר בין 0 ל-1. בהתאמה כאשר 0 = תשכח הכול, 1= תזכור הכול. אם נדבר לדוגמא על מודל שפה שמנסה לחזות את המילה הבאה על סמך כל המילים הקודמות. בבעיה כזו, ה-cell state עשוי לכלול את המין של הנושא הנוכחי, כדי שנוכל לחזות את המילים בהטיה הנכונה שלהם. אולם כאשר אנו רואים נושא חדש, אנו נרצה לשכוח את מין הנושא הישן. ניתן לסכם הכול לנוסחה מתמטית:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$

W_f מטריצת המצב, כפול המצב הנסתר הקודם והקלט הנוכחי פלוס biasה הנוכחי. כמובן שהכול נכנס לפונקציית הסיגמואיד. בצעד הבא נחליט איזה מידע הולך להיטען לתוך ה -cell state, לשלב זה יש שני חלקים. החלק הראשון הוא פונקציית הסיגמואיד שנקראת שער הקלט (input gate). שער זה מחליט איזה ערך לעדכן. בחלק השני נמצאת פונקציית tanh שיוצרת ערך 'מועמד' $\tilde{C}_t$ שיכול להיות שיוסף לcell state. אנחנו נשלב בין שני הערכים האלה כדי לעדכן ולייצר את הערך הסופי של המצב. בדוגמה הקודמת של מודל

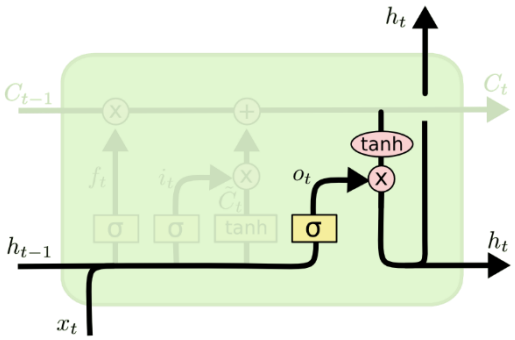
[9] Hochreiter, Sepp & Schmidhuber, Jürgen. (1997). Long Short-term Memory. Neural computation. 9. 1735-80. 10.1162/neco.1997.9.8.1735.

השפה שלנו, נרצה להוסיף את המגדר של הנושא החדש ל-cell state, כדי להחליף את הישן ששכחנו. נסתכל בנוסחאות שמתאימות להסבר ול**איור 27**:

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

$$\widetilde{C}_t = tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$$

בשלב הבא נעדכן את הcell state הישן - C_{t-1} למצב החדש - C_t. הצעדים הקודמים כבר החליטו מה עושים, מה שנותר הוא לבצע את החישובים האלה בפועל. לשם כך נכפיל את המצב הישן ב-f_t ובכך נשכח את מה שהוחלט להישכח. עכשיו נוסיף את הערך ה'מועמד' $\widetilde{C}_t$. כרגע יש לנו ערך 'מועמד' חדש כל שנותר הוא לבחור איזה ערך לעדכן ובכמה. במקרה של הדוגמא לעיל, זה המקום בו אנו בעצם נשמיט את המידע על מין הנושא הישן ונוסיף

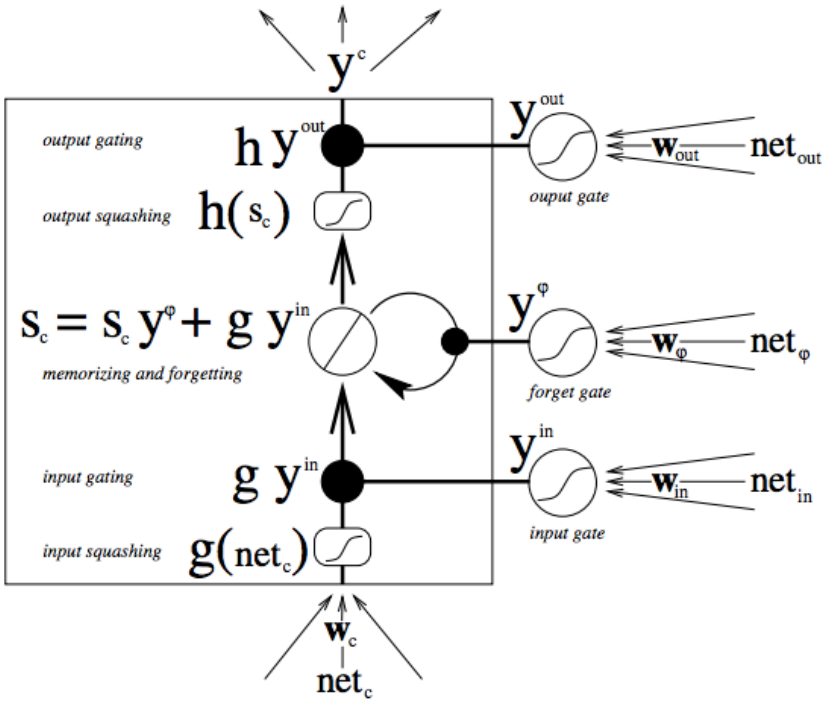


איור 29 הפלט בLSTM

את המידע החדש, כפי שהחלטנו בשלבים הקודמים. הנוסחאות שמתאימות להסבר ל**איור 28**:

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

לבסוף, עלינו להחליט מה יהיה הפלט. הפלט יתבסס על ה-cell state שלנו, אך זאת תהיה כבר גרסה מסוננת שלו. ראשית, אנו מפעילים שכבת sigmoid המחליטה אילו חלקים מה-cell state הולכים לפלט. לאחר מכן, הכנסנו את ה- cell state דרך tanh (כדי לדחוף את הערכים להיות בין 1- ל 1) נכפיל אותו בפלט של שער sigmoid, כך הפלט יהיה רק החלקים



איור 30 זרימת המידע דרך יחידת הזיכרון

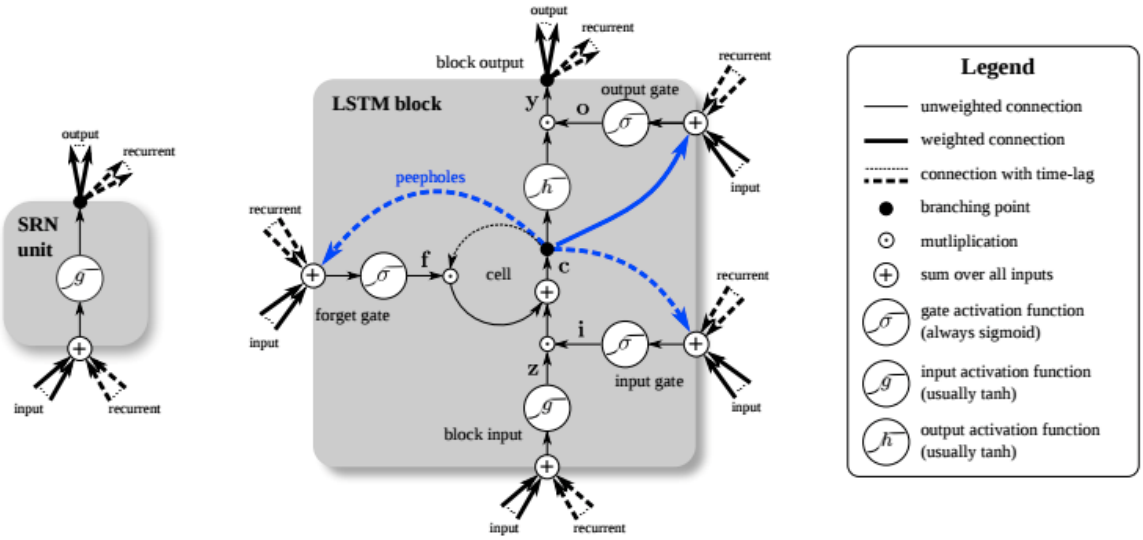
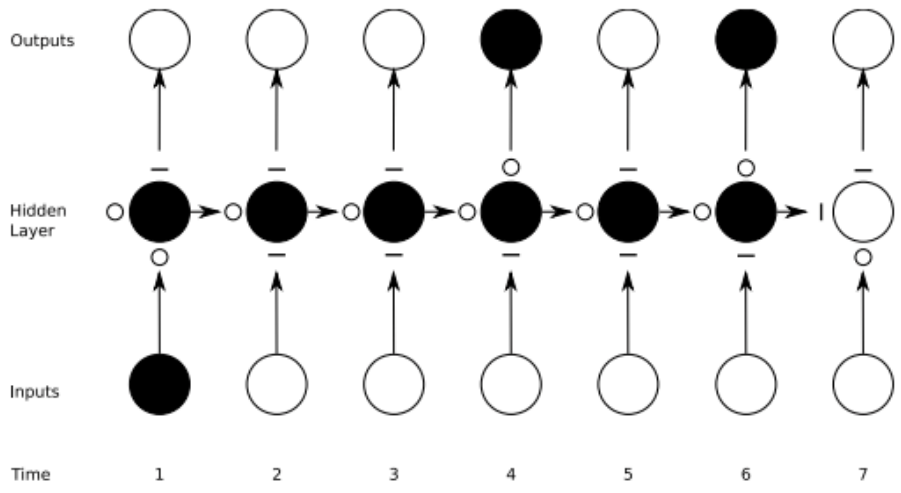


Figure 1. Detailed schematic of the Simple Recurrent Network (SRN) unit (left) and a Long Short-Term Memory block (right) as used in the hidden layers of a recurrent neural network.

איור 31 השוואה בין יחידת RNN פשוטה ליחידת LSTM מימין

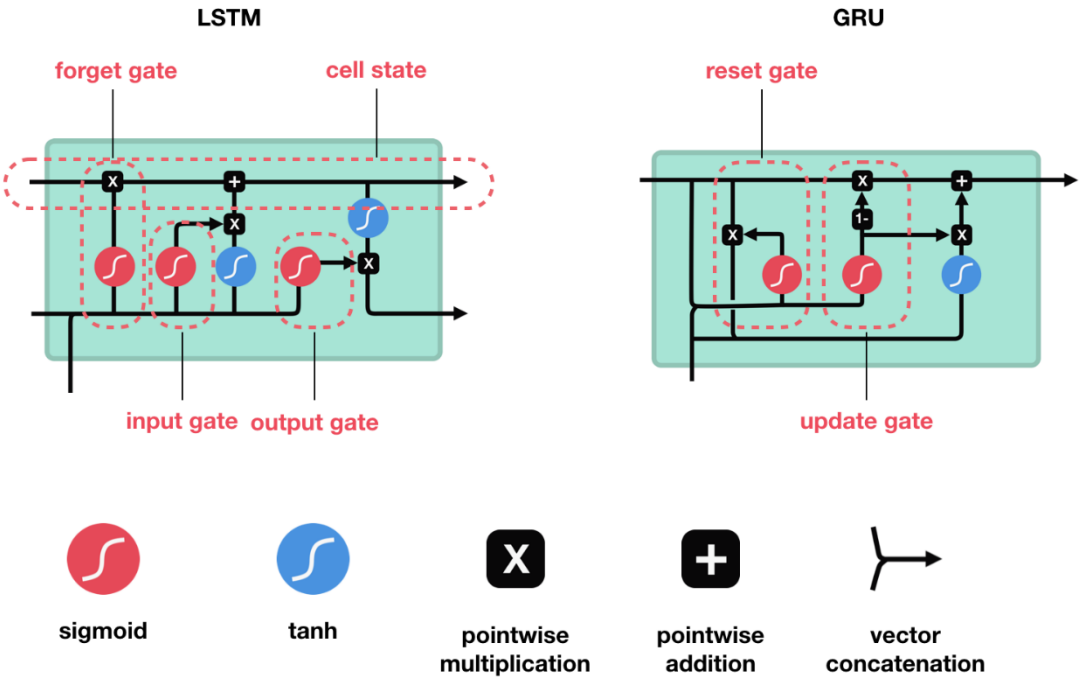
שהחלטנו עליהם. אם נחזור לדוגמא שלנו מכיוון שנניח שהרשת ראתה עכשיו נושא, ייתכן שהיא תרצה להוציא מידע רלוונטי עבור פועל (verb), למקרה שזאת תהיה המילה הבאה. נציע דרך הסבר אחרת - ב**איור 30** ניתן לראות את זרימת המידע דרך יחידת הזיכרון וכיצד מבנה היחידה והמעבר בשערים הייחודיים משפיע על המידע. נבהיר גם הפעם את האיור שלב אחרי שלב. נתחיל מלמטה: שלושת החיצים מראים את המידע זורם לתא. שילוב של קלט בהווה ומצב תא עבר (past cell state) לא רק לתא עצמו, אלא גם לכל אחד משלושת שעריו, אשר יחליטו כיצד יטופל הקלט. הנקודות השחורות הן השערים עצמם, שקובעים בהתאמה אם להכניס קלט חדש, למחוק את מצב התא הנוכחי ו / או לאפשר למצב זה להשפיע על פלט הרשת בשלב הנוכחי. S_c הוא המצב הנוכחי של תא הזיכרון, ו- $g * y^{in}$ הוא הקלט הנוכחי אל מצב זה. כאמור כל שער יכול להיות פתוח או סגור, והם ישלבו מחדש מצבים פתוחים וסגורים בכל צעד. התא בכל שלב בזמן יכול לשכוח את מצבו או לא; להיכתב או לא; ולקרוא, או לא (בצורה בינארית). האותיות הגדולות והמודגשות נותנות לנו את התוצאה של כל פעולה.



איור 32 המעבר ברשת של המידע

להלן איור נוסף - **איור 31** המשווה RNN פשוטה (משמאל) לתא LSTM (מימין). (אפשר להתעלם מהקווים הכחולים). חשוב לציין שתאי הזיכרון של LSTM מעניקים תפקידים שונים לתוספת ולכפל בשינויים שמתרחשים בקלט. סימן הפלוס המרכזי בשני התרשימים הוא למעשה הסוד של LSTMs. שינוי זה נראה פשוט ובסיסי אבל שינוי בסיסי זה מסייע להם לשמור על שגיאה קבועה ומתמדת כאשר מתבצע תהליך backpropagation. במקום לקבוע את מצב התא שלאחר מכן על ידי הכפלת מצבו הנוכחי עם קלט חדש, אנחנו *מוסיפים* גם את המצב הנוכחי וזה כל ההבדל. (יש לציין כי שער השכחה עדיין מסתמך על כפל, כמובן.) קבוצות משקולות שונות מסננות את הקלט עבור קלט, פלט ושכחה. שער השכחה מיוצג כפונקציה של זהות לינארית, מכיוון שאם השער פתוח, המצב הנוכחי של תא הזיכרון פשוט מוכפל אחד, כדי להפיץ קדימה צעד נוסף פעם נוספת. קבוצות משקולות שונות מסננות את הקלט עבור מידע שישלח אל מעברי הקלט, פלט ושכחה. שער השכחה

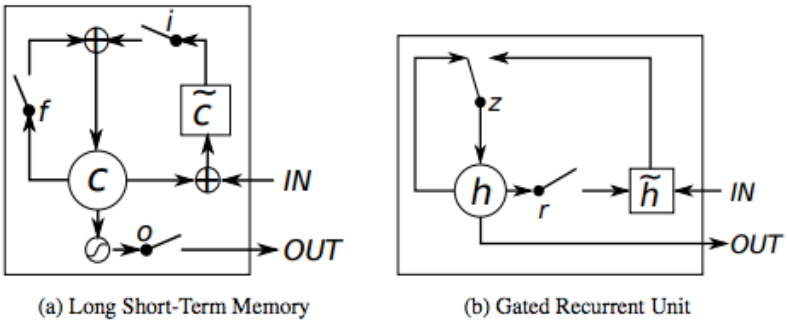
מיוצג כפונקציה של זהות[10] לינארית, מכיוון שאם השער פתוח, המצב הנוכחי של תא הזיכרון פשוט מוכפל באחד, כדי להפיץ (propagate) הלאה צעד נוסף בזמן (time (step. שאלה מתבקשת היא - מדוע ל- LSTMs יש שער שכחה כאשר מטרתם לקשר אירועים מרוחקים לתפוקה סופית?. ובכן, לפעמים פשוט טוב לשכוח. לדוגמא אם אנחנו מנתחים קורפוס של טקסט ומגיע סוף מסמך, למשל, יתכן שאין לנו סיבה להאמין שלמסמך הבא יש קשר כלשהו למסמך הנוכחי, ולכן יש לאפס את תא הזיכרון לפני שהרשת ניגשת אל המרכיב הראשון של המסמך הבא. **באיור 32**, ניתן לראות את השערים בזמן פעולה, כאשר קווים ישרים מייצגים שערים סגורים, ומעגלים ריקים מייצגים שערים פתוחים. הקווים והמעגלים העוברים אופקית במורד השכבה הנסתרת הם שערי השכחה. ראוי להזכיר כי בעוד שרשתות העדכונים ממפות קלט אחד לפלט אחד, RNN יכולות למפות קלט אחד לרבים, כנ"ל (תמונה אחת למספר מילים בכיתוב), רבות לרבים (תרגום) או רבות לאחת (סיווג קול). השאלה היא מהו הערך המדויק של שערי קלט המגנים על תא זיכרון מפני כניסת נתונים חדשים, ושערי פלט שמונעים ממנו להשפיע על פלטים מסוימים של RNN. אתה יכול לחשוב על LSTMs כמאפשרים לרשת נוירונים לפעול בסכמות זמן שונות בבת אחת. לשם הדוגמא ניקח פרק זמן ממשך חיי אדם, ונתאר לעצמנו שאנחנו מקבלים כמויות שונות של נתונים על החיים של אותו אדם בסדר כרונולוגי. נניח שהמיקום הגיאוגרפי בכל



איור 33 GRU vs LSTM

שלב בזמן חשוב למדי לשלב הפעם הבא, כך שסולם הזמן תמיד פתוח למידע העדכני ביותר. אם האדם הזה הוא אזרח חרוץ שמצביע כל שנתיים (או פעמיים בשנה אם הוא אזרח ישראלי) ונרצה לחזות את הבחירה שלו. בזמן דמוקרטי, נרצה לשים לב במיוחד לאירועים

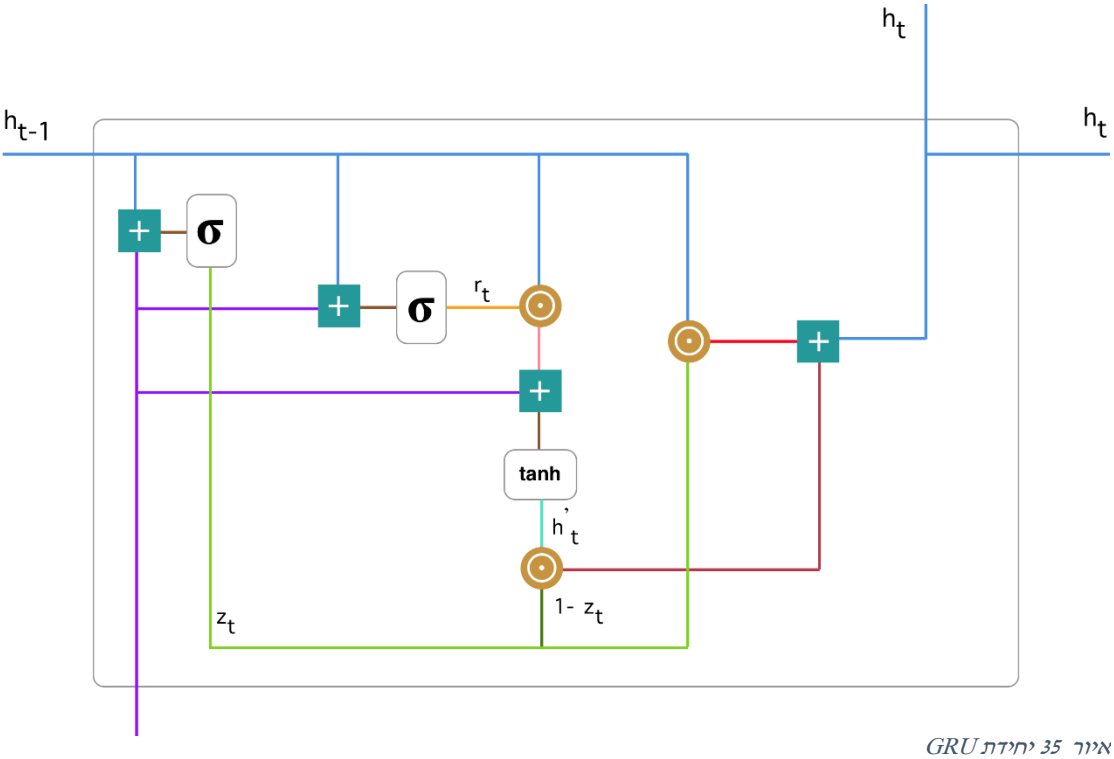
[10] פונקציית הזהות או טרנספורמציית הזהות היא פונקציה שמחזירה תמיד את אותו הערך שעליו היא פעלה, פונקציה f היא פונקציית הזהות אם לכל איבר x בקבוצה M עליה היא פועלת מתקיים { f(x)=x }.



(a) Long Short-Term Memory (b) Gated Recurrent Unit

Figure 1: Illustration of (a) LSTM and (b) gated recurrent units. (a) i, f and o are the input, forget and output gates, respectively. c and $\tilde{c}$ denote the memory cell and the new memory cell content. (b) r and z are the reset and update gates, and h and $\tilde{h}$ are the activation and the candidate activation.

איור 34 LSTM vs GRU



איור 35 יחידת GRU

שמתרחשים לפני קיום הבחירות, לפני שהם יחזרו לשגרה, ויתרחקו מנושאים גדולים יותר. מצד שני אנחנו לא נרצה לתת לרעש המתמיד של מיקום גיאוגרפי להשפיע על הניתוח הפוליטי שלנו. אם לאותו אדם יש גם בת חרוצה, אז אולי נוכל לבנות זמן משפחתי שלומד דפוסים בשיחות טלפון שמתקיימות באופן קבוע בכל יום ראשון ומשתנה מדי שנה סביב החגים.

4.1.6.3 Gated Recurrent Units (GRUs)

Gated Recurrent Units (GRU) היא בעצם LSTM ללא שער פלט, ולכן היא כותבת את התוכן במלואו מתא הזיכרון שלו להמשך הרשת בכל שלב. כמתואר **באיור 33 ואיור 34.** כמו LSTM גם GRU נוצר כדי להתמודד עם בעיית ה-vanishing gradient . במקרים

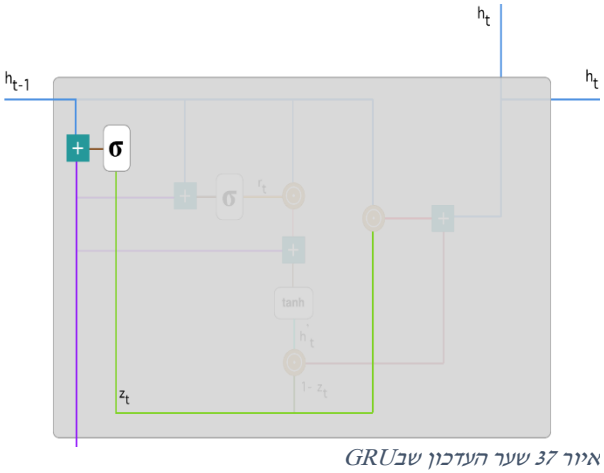
איור 36 משמעות בסימנים שבאיור 35

רבים GRU ו-LSTM מאוד דומים בתוצאות, וזה יהיה חלק מן המחקר שלנו להשוות בין שתי צורות של RNN עבור משפטים ארוכים. נבחן את המודל שלנו עבור GRU ועבור LSTM. כאמור כדי לפתור את בעיית vanishing gradient של RNN סטנדרטית, GRU משתמש, במה שנקרא, update gate and reset gate. בעיקרון, מדובר בשני ווקטורים המחליטים איזה מידע צריך להעביר לפלט. הדבר המיוחד בהם הוא שניתן לאמן אותם לשמירת מידע ישן, מבלי שיעלם לאורך זמן. כמו כן יש להם את היכולת להסיר מידע שאינו רלוונטי לתחזית. בכדי להסביר את המתמטיקה שמאחורי התהליך נבחן יחידה אחת המופיעה ב**איור 35**. **באיור 36** ניתן לראות את משמעות הסימנים שקיימים ב**איור 35.**

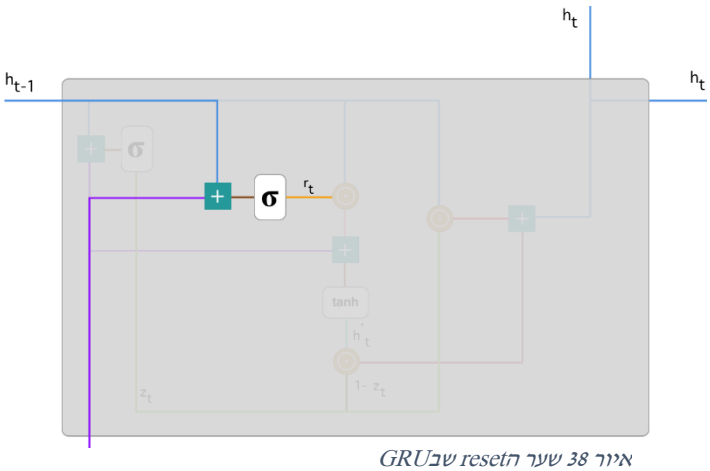
נתחיל בהסבר על update gate: נתחיל בחישוב *שער העדכון* z_t *לשלב זמן* t באמצעות הנוסחה:

$$Z_t = \sigma(W^{(z)}x_t + U^{(z)}h_{t-1})$$

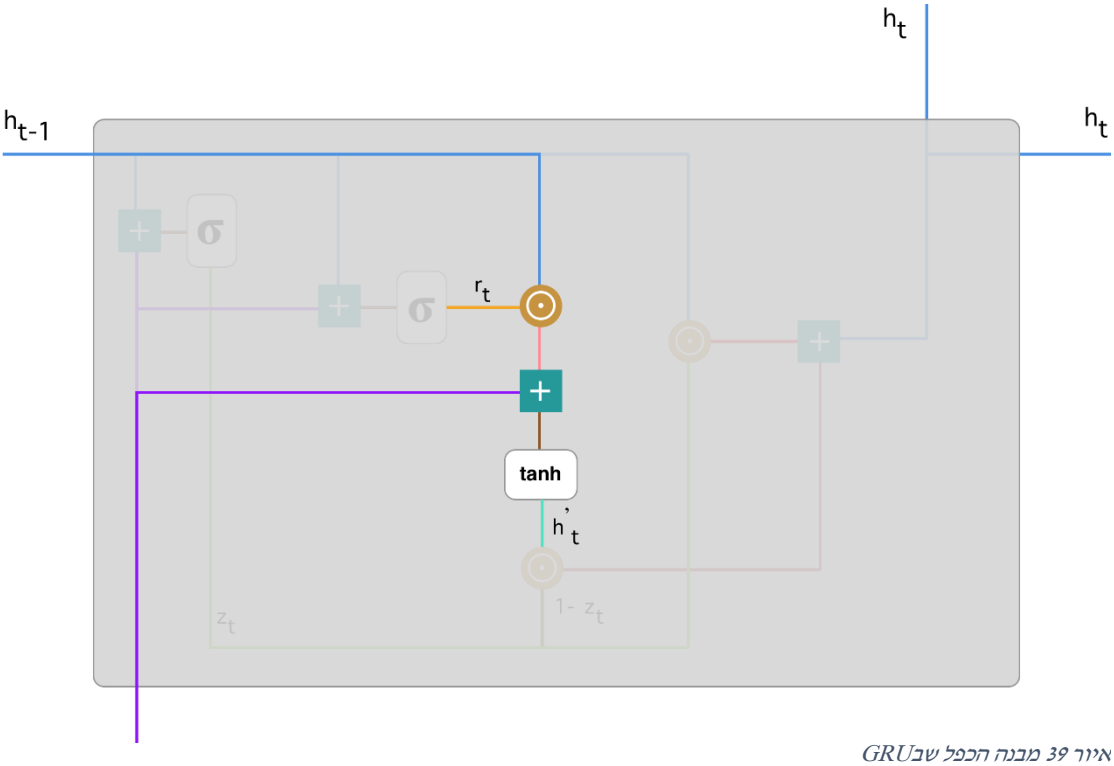
כאשר x_t הוא הקלט של היחידה ברשת, מכפילים את הקלט כפול $W^{(z)}$. אותו דבר עבור h_{t-1} שמכיל את המידע מהיחידות הקדומות. כפול ווקטור המשקלות שלו - $U^{(z)}$. שתי התוצאות מחוברות יחד ונכנסות לתוך פונקציית הסיגמואיד כך שיכווץ לערך בין 0 ל-1. המיקום הוא כמתואר **באיור 37**. שער העדכון (update gate) מסייע למודל לקבוע כמה צריך להעביר ממידע העבר (מצעדים קודמים) לעתיד. זה באמת עוצמתי מכיוון שהמודל יכול להחליט להעתיק את כל המידע מהעבר ולחסל את הסיכון לבעיית vanishing gradient. נראה את השימוש בשער העדכונים בהמשך. לעת עתה מספיק לזכור את



*איור 37 שער העדכון שב*GRU



*איור 38 שער הreset שב*GRU



איור 39 מבנה הכפל שב-GRU

הנוסחה עבור Z_t. השער השני מכונה Reset gate: בעיקרון, שער זה משמש כדי להחליט כמה מהמידע לשכוח. בשביל לחשב את זה אנו משתמשים בנוסחה הבאה:

$$r_t = \sigma(W^{(r)}x_t + U^{(r)}h_{t-1})$$

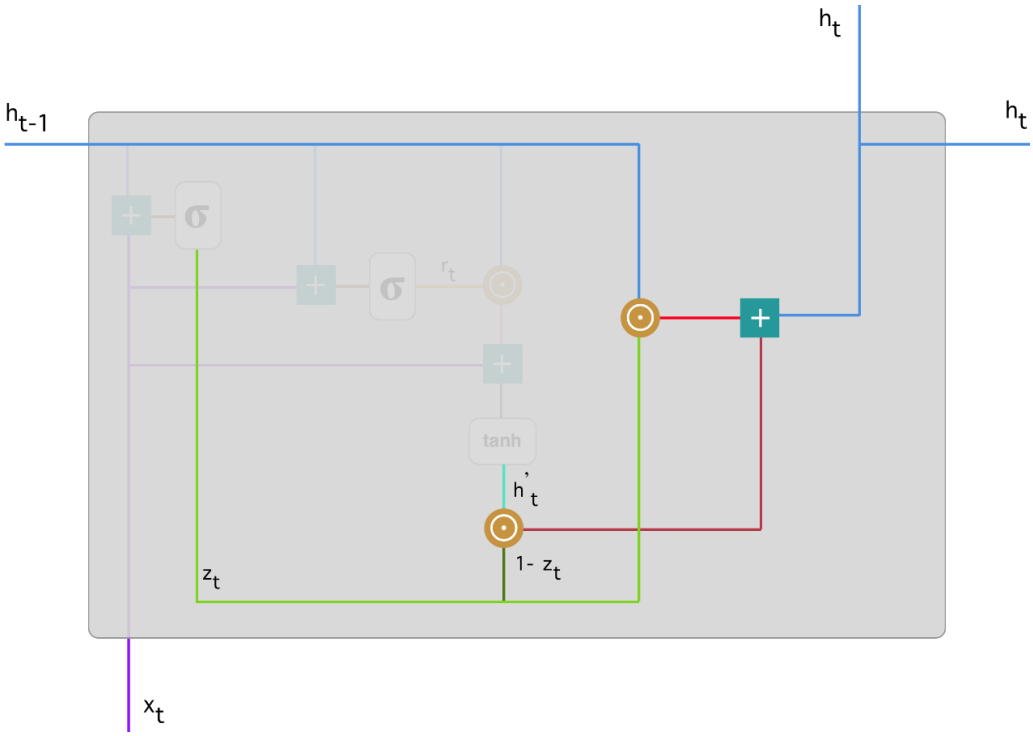
נוסחה זו זהה לזו של שער העדכון. ההבדל מגיע במשקולות ובשימוש בשערים. **באיור 38** ניתן לראות היכן נמצא שער האיפוס. כמו בפעם קודמת אנו מחברים את h_{t-1} - קו כחול ו- x_t - קו סגול, מכפילים אותם עם המשקלים המתאימים שלהם, מסכמים את התוצאות ומפעילים על הסכום את פונקציית sigmoid. תוכן התא הנוכחי[11]: כרגע נבחן כיצד השערים שהוזכרו לעיל ישפיעו על הפלט הסופי. ראשית נדבר על השימוש של שער האיפוס.

[11] תוכן התא הנוכחי מוגדר כשלב ביניים בדרך לפלטים של כל היחידה וסומן כ-h'.

לצד שער האיפוס קיימת סדרה של פעולות שמטרתן לעזור לשער האיפוס לבחור את המידע הרלוונטי שהגיע מהעבר וכרגע נדרש לשימוש. נחשב את המשימה הזאת בצורה הבאה:

$$h'_t = \tanh\left(Wx_t + r_t \odot Uh_{t-1}\right)$$

1. נבצע כפל של x_t (הקלט הנוכחי) עם המשקל הנוכחי. נבצע את אותן פעולות עבור המצב הנסתר הקודם $h_{(t-1)}$ עם מטריצת המשקלות הקודמים U.



איור 40 החלק הסופי של הGRU

2. נבצע כפל בשיטת אדמר [12] בין הפלט של שער האיפוס r_t ומטריצת המשקלים הקודמים (אם נחשוב על זה, זאת בעצם עיקר הפעולה שמבצעת את המשימה שלנו: לקיחת תוכן רלוונטי מהעבר בהתאם לצורך שמחושב בשער האיפוס). נניח שיש לנו בעיית ניתוח רגשות לקביעת דעתו של מישהו על ספר מתוך ביקורת שכתב. הטקסט מתחיל ב"זהו ספר פנטזיה שממחיש ... "ואחרי כמה פסקאות מסתיים ב"לא ממש נהניתי מהספר כי אני חושב שהוא לוכד יותר מדי פרטים." כדי לקבוע את רמת הסיפוק הכללית מהספר אנו זקוקים רק לחלק האחרון של הביקורת. במקרה זה רשת

[12] כפל בשיטת אדמר מתבצע כך:

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \circ \begin{bmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{bmatrix} = \begin{bmatrix} a_{11}\,b_{11} & a_{12}\,b_{12} & a_{13}\,b_{13} \\ a_{21}\,b_{21} & a_{22}\,b_{22} & a_{23}\,b_{23} \\ a_{31}\,b_{31} & a_{32}\,b_{32} & a_{33}\,b_{33} \end{bmatrix}$$

הנוירונים תיגש לסוף הטקסט ותלמד לתת ערך נכון לווקטור ה– r_t בכך היא יוצרת שכחה של התחלת הטקסט והתמקדות בסוף הרלוונטי.

3. חיבור של תוצאות שלב 1 ו-2.
4. ביצוע פונקציית האקטיבציה tanh.

ניתן לראות את המבנה **באיור 39**. בעצם ניתן לראות את הכפל שמתבצע בין הקו הכחול (h_{t-1}) והקו הכתום (r_t) תוצאת החישוב מוצגת בצבע ורוד. הפלט בצבע סגול $x_t)$)ופונקציית האקטיבציה היא $tanh$ שמייצרת את המצב h'_t בצבע ירוק בהיר. <u>הזיכרון הסופי בזמן הנוכחי:</u> בצעד האחרון של הרשת, דרוש לחשב את ה- h_t כאמור זהו הווקטור שמחזיק את המידע עבור כל יחידה, וכמובן מועבר הלאה בהתאם לחישובים. כדי להעביר מידע יש צורך בפלט של שער העדכון. שער העדכון אחראי על כמות המידע שיש לאסוף מתוך h'_t וכמות המידע מתוך h_{t-1}. החישוב יתבצע כך:

TABLE V: **Semantic Role Labeling**

Paper	Model	CoNLL2005 (F1 %)	CoNLL2012 (F1 %)
Collobert et al.	CNN with parsing features	76.06	
Täckström et al.	Manual features with DP for inference	78.6	79.4
Zhou and Xu	Bidirectional LSTM	81.07	81.27
He et al.	Bidirectional LSTM with highway connections	83.2	83.4

איור 41 השוואות בתוצאות עבור תיוג סמנטי

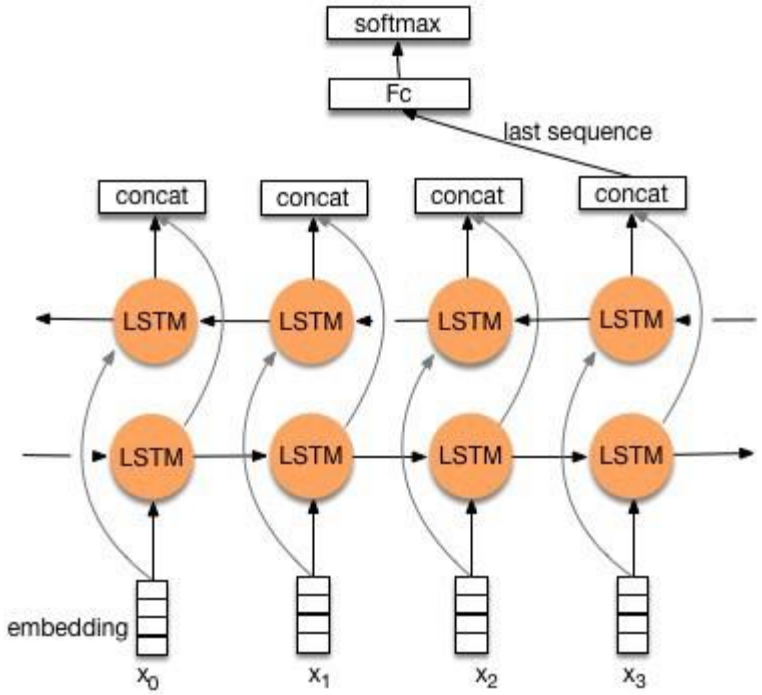
$$h_t = z_t \odot h_{t-1} + (1 - z_t) \odot h'_t$$

נזכר בדוגמא לגבי ביקורת הספרים. הפעם, המידע הרלוונטי ביותר ממוקם בתחילת הטקסט. המודל יכול ללמוד להגדיר את הווקטור z_t קרוב ל -1 ולשמור על רוב המידע

Table 3. The result of the sequence model classifier performance.

Training: Testing (%)	Sequence Model Classifier	Performance Metrics (%)					
		Sensitivity	Specificity	Precision	F1-score	BACC	MCC
90:10	Vanilla RNN	85.81	87.92	89.56	84.97	88.14	89.85
	LSTM	98.49	97.97	95.67	96.32	97.56	95.32
	GRU	87.07	98.10	94.89	94.08	98.73	93.78
80:20	Vanilla RNN	86.86	87.28	88.37	82.40	81.66	89.64
	LSTM	92.47	97.62	90.11	88.57	89.81	79.62
	GRU	87.17	88.49	90.60	86.69	88.90	87.90
70:30	Vanilla RNN	81.88	90.78	60.46	63.60	75.00	67.08
	LSTM	88.18	93.61	71.51	82.55	83.33	78.78
	GRU	92.59	93.60	71.51	82.27	83.33	78.78
60:40	Vanilla RNN	67.09	91.12	60.46	69.56	75.00	67.08
	LSTM	97.61	92.16	65.11	74.91	75.00	67.08
	GRU	96.85	93.79	72.67	81.43	83.33	78.78
50:50	Vanilla RNN	31.44	88.70	64.53	42.28	73.14	54.71
	LSTM	88.14	93.00	69.18	77.52	83.33	78.78
	GRU	51.02	87.80	43.41	46.91	67.06	48.50

איור 42 השוואת בתוצאות בין רשתות שונות



איור 43 מבנה סופי של LSTM עם softmax

הקודם. מכיוון ש z_t יהיה קרוב ל 1 בשלב זה, $1 - z_t$ יהיה קרוב ל 0 מה שיגרום למערכת להתעלם מחלק גדול מהתוכן הנוכחי (במקרה זה החלק האחרון של הביקורת שמסביר את עלילת הספר) שאינו רלוונטי עבורנו חיזוי. כמתואר **באיור 40**. באיור תוכל לראות כיצד z_t-קו ירוק משמש לחישוב $1 - z_t$, אשר בשילוב עם h'_t קו ירוק בהיר, מניב תוצאה בקו האדום הכהה. z_t משמש גם עם h_{t-1} קו כחול במכפלת אדמר. לבסוף, קו h_t - כחול הוא תוצאה של סיכום הפלטים המתאימים לקווים האדומים הבהירים והכהים.

תוצאות לדוגמא: באיורים 41 ו-42 ניתן לראות תוצאות של משימות NLP ע"י שימוש ברשתות נוירונים שונות.

4.1.6.4 Bidirectional RNN

רשתות עצביות דו-כיווניות חוזרות (RNN) הן בסה"כ חיבור של שתי RNNs עצמאיות. כאשר רצף הקלט (במקרה שלנו משפט) מוזן לפי סדר זמן רגיל לרשת אחת, ובסדר זמן הפוך לרשת האחרת. הפלטים של שתי הרשתות בדרך כלל משולבות בכל שלב בזמן, אם כי לעיתים משלבים בצורות שונות, למשל במשימות של סיכום טקסט וכדו'. מבנה זה מאפשר לרשתות לקבל מידע אחורה וקדימה על הרצף בכל שלב. הרעיון נראה די קל אבל כשמדובר ביישום זה קצת הופך להיות מורכב. הבעיה הראשון היא לגבי הדרך להעביר את פלטי ה-RNN הדו כיווני לרשת נוירונים צפופה. לגבי RNN רגילים נוכל פשוט להעביר את הפלטים בשלב האחרון, אבל בפלט דו כיווני אם נבחר פשוט את הפלט בשלב האחרון, ה- RNN ההפוך יראה רק את הקלט האחרון (X_3 **באיור 43**). התוצאה כמעט ולא תספק שום כוח חזוי. הבעיה השנייה היא סביב המצבים הנסתרים החוזרים (returned hidden states).

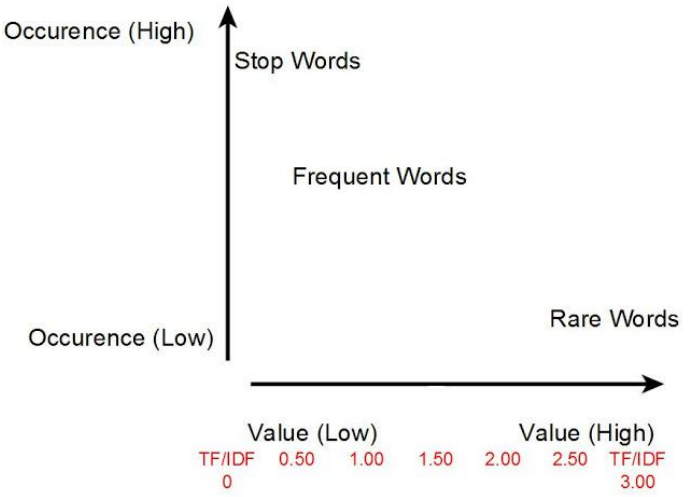
במודלים מסוג seq2seq (יוסבר בהמשך), נרצה כי מצבים נסתרים מהמקודד יאתחלו את המצבים הנסתרים של המפענח. באופן אינטואיטיבי, אם נוכל לבחור רק במצבים נסתרים בשלב מסוים, נרצה שהמצב בו ה- RNN פשוט ייקח את הקלט האחרון ברצף.

4.2 Word Embedding הטמעת מילים

הגדרה בסיסית מאוד של הטמעת מילים היא מספר (real number) או ייצוג וקטורי של מילה. בדרך כלל ננסה שלמילים עם משמעות דומה יהיו ייצוגים וקטוריים הקרובים זה לזה במרחב ההטמעה (אם כי זה לא תמיד היה המקרה). המטרה בבניית מילה והטמעה במרחב, היא 'להשיג' סוג כלשהו של הקשר בתוך אותו מרחב. יהיה זה משמעות, מורפולוגיה או סוג אחר של מערכת יחסים. על ידי קידוד מילים במרחב המאוכלס בצפיפות הפרוש על מספר ממדים אנו יכולים לייצג מילים בצורה מספרית באופן שנציג אותם בווקטורים בעלי עשרות או מאות ממדים במקום מיליונים (כמו One-hot-vector יוסבר בהמשך). הטמעות רבות של מילים נוצרות על בסיס התפיסה שהוצגה על ידי "השערת ההפצה[13]" של זליגל האריס, המבטאת את הרעיון הפשוט שלמילים המשמשות קרוב זו לזו בדרך כלל יהיה את אותה משמעות. היופי בתוצרים של הליכה בשיטתו של זליגל, הוא שהטמעת מילים שונות הנוצרות באופנים שונים או באמצעות תאגידי טקסט שונים כדי למפות את מערכת היחסים הזו, מקבלים ערכים דומים ולכן התוצאה הסופית עוזרת לנו במשימות שונות בעולם ה- NLP. מילים אינן דברים שמחשבים מבינים באופן טבעי. על ידי קידודם בצורה מספרית, נוכל להחיל כללים מתמטיים ולבצע עליהם פעולות או להופכם למטריצה. יכולת זו הופכת אותם לשימושיים במיוחד בעולם למידת המכונה.

אם ניקח לדוגמא למידה עמוקה (deep learning). על ידי קידוד מילים בצורה מספרית, אנו יכולים לקחת ארכיטקטורות רבות בתחום וליישם אותן על מילים או משפטים. ראינו בפרק הקודם דוגמאות של רשתות נוירונים שהוחלו על משימות NLP תוך שימוש בהטמעת מילים והביאו את הביצועים הטובים ביותר עבור משימות רבות. למעשה הרעיון הזה של הטמעת מילים מביא לכך שלא צריך ללמוד קבוצה חדשה של הטמעות לכל משימה, או לכל מודל. במקום זאת, אנו יכולים ללמוד ייצוג כללי אשר ניתן להשתמש בו במשימות שונות.

[13] The **Distributional Hypothesis** is that words that occur in the same contexts tend to have similar meanings (Harris, 1954). The underlying idea that "a word is characterized by the company it keeps" was popularized by Firth (1957), and it is implicit in Weaver's (1955) discussion of word sense disambiguation (originally written as a memorandum, in 1949). The Distributional Hypothesis is the basis for Statistical Semantics. Although the Distributional Hypothesis originated in Linguistics, it is now receiving attention in Cognitive Science (McDonald and Ramscar, 2001). The origin and theoretical basis of the Distributional Hypothesis is discussed by Sahlgren (2008).



איור 44 tf-idf

בדרך אגב ניתן לומר שרעיון זה של הטמעת מילים הוא נגזרת ישירה של מסקנות ופילוסופיות של השפה. כפי שראינו בפרק 3 - ייצוג סימבולי של משמעות וכן 'משל המערה של אפלטון'. עכשיו לאחר הקדמה קצרה זו, נציג בקצרה כמה מהדרכים השונות בהן אנו יכולים לייצג מילים באופן כללי.

4.2.1 One-Hot Encoding

אחת הדרכים הבסיסיות ביותר שאנו יכולים לייצג מילים היא באמצעות שיטת הקידוד one-hot encoding (נקראת לעיתים גם count vectorizing). הרעיון של שיטה זו פשוט מאוד. לכל מילה ניצור וקטור בעל מידות רבות ככל שלקובץ הטקסט שאנו עובדים עליו (קורפוס) יש מילים ייחודיות. לכל מילה ייחודית יש מימד ייחודי והיא תיוצג על ידי הערך 1 בממד הזה עם 0 בכל שאר הממדים. התוצאה של שיטה זו היא וקטורים ענקיים ודלילים שתופסים שום מידע עם היבט יחסי. להשתמש בשיטה זו כדאי במקרים של טקסטים קצרים או כאשר אין אפשרות אחרת. במחקר שלנו אנו זקוקים למידע עם קשר סמנטי ולכן נעדיף להשתמש בשיטות מתאימות.

4.2.2 TF-IDF Transform

וקטורי TF-IDF דומים לווקטורים מסוג one-hat. עם זאת, במקום להציג רק עם ערך אחד בווקטור הייצוג, הם מציגים ייצוגים מספריים שבהם מילים מופיעות. כלומר, **מילים מיוצגות על ידי התדירות שלהן כפול התדירות ההפוכה שלהן בכל הטקסט עצמו.** כפי שניתן לראות לדוגמא **באיור 44** במונחים פשוטים יותר, מילים המופיעות הרבה ובכל מקום יש לתת להם מעט מאוד משקל או משמעות. אנו יכולים לקחת כדוגמא את המילים "אני", "וגם", "הוא" בשפה העברית, ניתן לראות שהם לא מספקים כמות גדולה של ערך משמעותי. עם זאת, אם מילה מופיעה מעט מאוד או שהיא מופיעה בתדירות גבוהה אך ורק במקומות ספורים, הסברא אומרת שמדובר במילים בעלות חשיבות גבוהה יותר ויש לשקלל אותן ככאלה. שוב, טכניקה זו גם סובלת מהחיסרון של ייצוגים בממדים גבוהים מאוד שאינם תופסים קשר סמנטי כמו ב-one hot. טכניקה זו פותחה במקור כמדד לדירוג של פונקציות להצגת תוצאות מנועי חיפוש בהתבסס על

	Roses	are	red	Sky	is	blue
Roses	1	1	1	0	0	0
are	1	1	1	0	0	0
red	1	1	1	0	0	0
Sky	0	0	0	1	1	1
is	0	0	0	1	1	1
Blue	0	0	0	1	1	1

איור 45 מטריצת עיבוד למילים

שאילתות משתמשים והפכה להיות חלק מאחזור מידע וחילוץ תכונות טקסט. כעת נגדיר פורמאלית את TF-IDF.

TF= (מספר הפעמים שהמילה מתרחשת בטקסט) / (המספר הכולל של המילים בטקסט).

IDF= (המספר הכולל של המסמכים / מספר המסמכים עם המילה t)

והנוסחה הסופית:

$$\mathrm{TF-IDF} = \mathrm{TF} * \mathrm{IDF}$$

בפועל משתמשים בנוסחה הבאה[14]:

$$w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right)$$

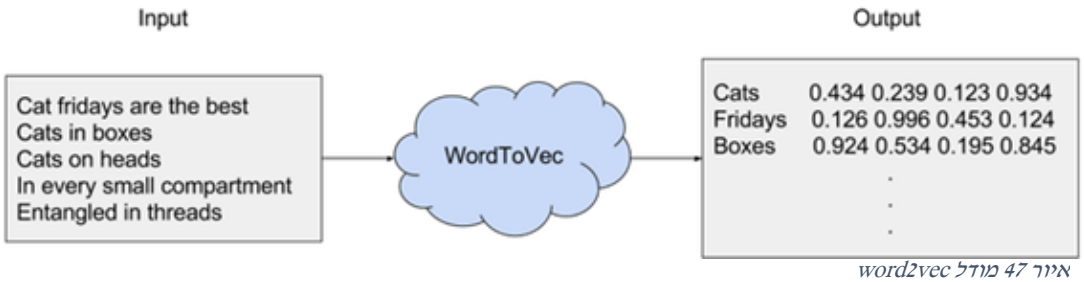
4.2.3 Co-Occurrence Matrix

מטריצה משותפת היא מטריצה ענקית כאשר מספר השורות והעמודות הוא כגודל אוצר המילים שיש בכל הקורפוס. אם מילים מופיעות יחד (כלומר קרוב אחת לשנייה), הן מסומנות בערך חיובי. אחרת, 0. זה מסתכם בייצוג מספרי שפשוט שואל את השאלה: האם מילים מופיעות יחד? אם כן, אז תסמן את זה. הבעיה העיקרית היא שבכך נוצר ייצוג מאוד גדול, אם חשבנו ש-One hot הוא בעל מספר ממדים גבוה, אז מטריצת התרחשות משותפת היא גבוהה בריבוע. בכך נוצרת בעיה של זיכרון כי יש הרבה נתונים שצריך לאחסן. במטריצה ש**באיור 45**, כל תא מציין אם שני הפריטים 'מתרחשים' יחד או לא. ניתן גם להחליף את המספר הזה במספר הפעמים שהן מתרחשות, או לחילופין בגישה מתוחכמת יותר. אפשר גם לשנות את הישויות עצמן, על ידי הצבת שמות עצם בעמודות ותיאורים בשורות במקום כל מילה. השימוש הבולט בשיטה זו ב-NLP הוא היכולת לספק יחסים בין מושגים. לשם הדוגמא נניח שאנחנו עובדים על ביקורות של מוצרים, ונניח בפשטות שכל סקירה מורכבת רק ממשפטים קצרים. יהיה לנו משהו כזה:

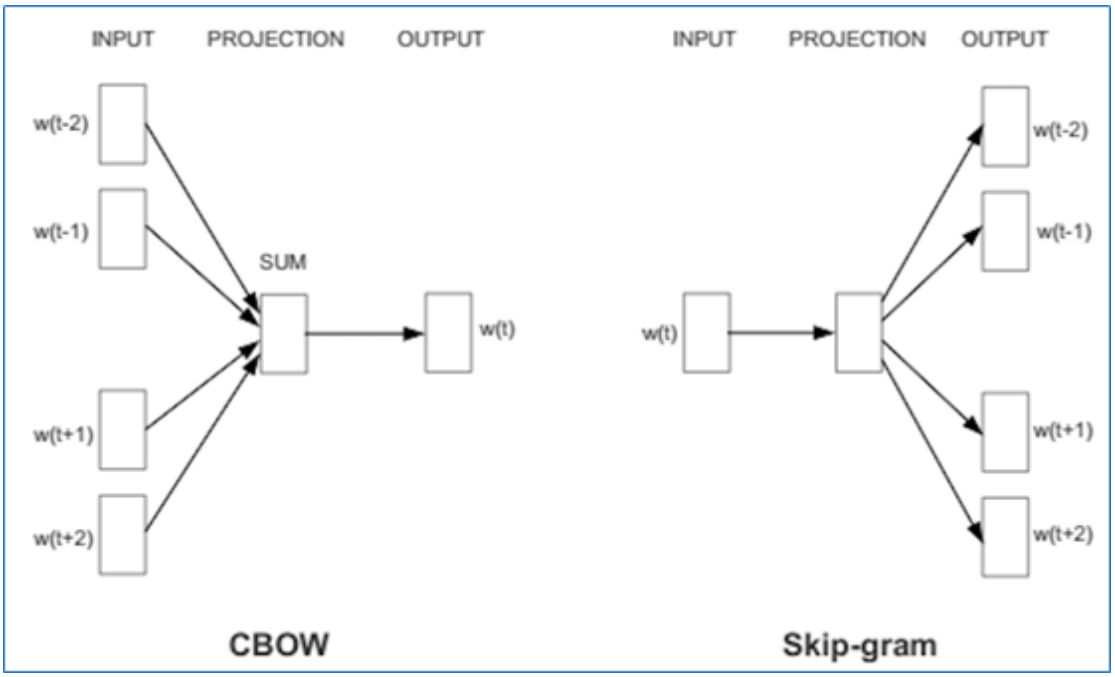
מוצר א' הוא מדהים.

אני שונא את מוצר ב'.

[14] הסיבה ארוכה מידי, ניתן לראות כאן https://mailman.uib.no/public/corpora/2018-June/thread.html



איור 47 מודל word2vec



איור 46 skip-gram vs CBOW

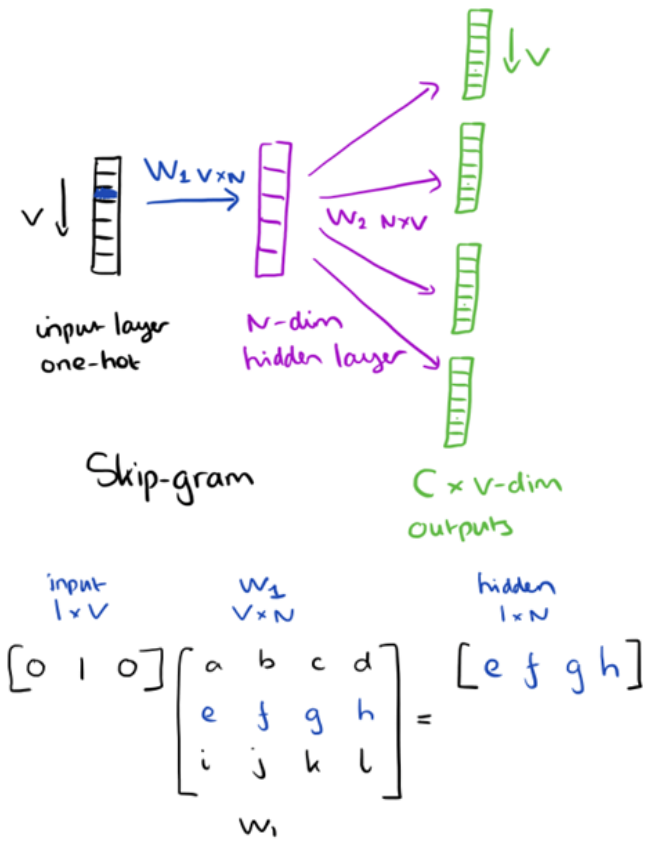
ייצוג ביקורות אלו בקלות כמטריצה אחת להתרחשות משותפת יאפשר לשייך מוצרים לפי ההערכה.

4.2.4 Word2vec

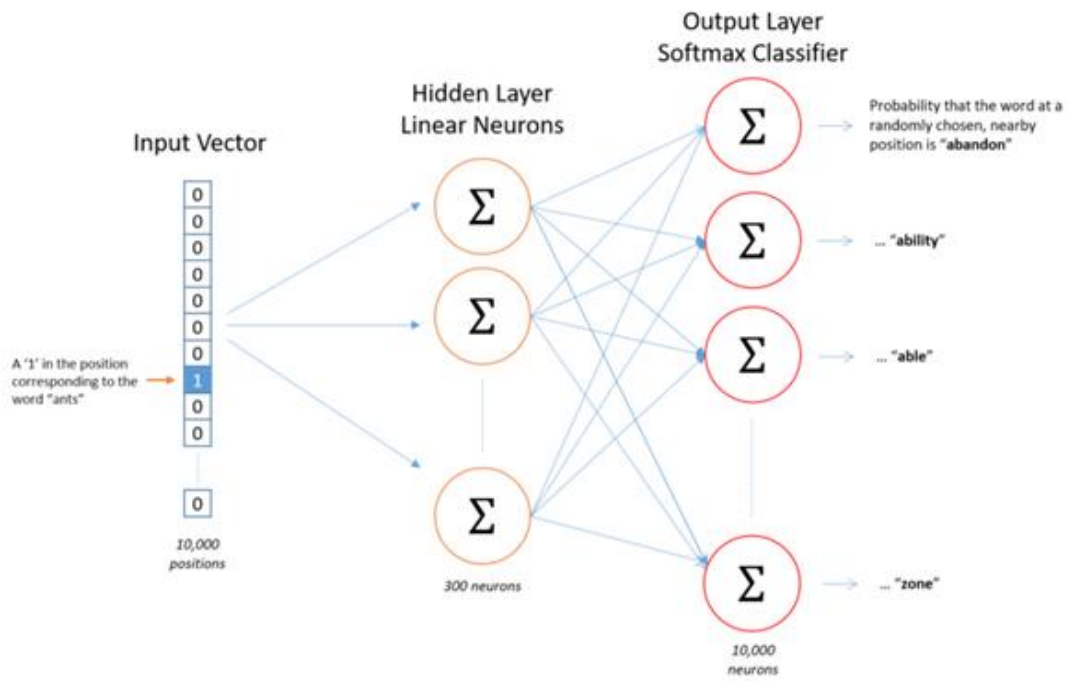
Word2vec הוא שילוב של מודלים המשמשים לייצוג ייצוגים מבוזרים של מילים בקורפוס מסוים. Word2Vec (W2V) הוא אלגוריתם שמקבל קורפוס טקסט כקלט ומייצר ייצוג וקטורי לכל מילה, כפי שמוצג ב**איור 46**. ישנם שני מימושים של האלגוריתם הזה, CBOW ו- Skip-Gram. בהינתן קורפוס המודל מריץ עיבוד בלולאה על המילים של כל משפט כאשר הוא משתמש במילה הנוכחית w כדי לחזות את שכניה (כלומר, ההקשר שלה), גישה זו מכונה "**skip-gram**". דרך אחרת לבצע את העיבוד היא להשתמש בכל אחד מההקשרים האלה השכנים למילה w כדי לחזות את המילה הנוכחית w, במקרה זה השיטה נקראת "**Continuous Bag Of Words**" (CBOW). כדי להגביל את מספר המילים שנתחשב בהם כדי לקבל הקשר, משתמשים בפרמטר שנקרא "גודל החלון". הווקטורים שאנו משתמשים בהם כדי לייצג מילים בשיטה זו נקראים הטמעת מילים עצבית (neural word embeddings), מכיוון שבשיטה זו נשתמש ברשת נוירונים בעלת שכבה אחת כדי לייצר את הייצוג של

איור 48 דוגמא לword2vec

המילים. ב**איור 47** ניתן לראות את ההבדל בין שתי השיטות. ההקשר של כל מילה בקורפוס הוא שקובע את הערך של המילה. המודל מתאמן על המילה בהתחשב בהקשרים הסביבתיים שלה ומייצר את הווקטור המתאים עבורה. וקטורי המילים שנמצאו דומים לפי הקשרם מיוצגים קרוב זה לזה על ידי התאמת המספרים בווקטור. במחקר זה, אנו הולכים להתמקד במודל Skip-Gram שבניגוד ל- CBOW רואה במילה הנוכחית את שאר המילים הקרובות אליה כפי שמתואר באיור למעלה. השאלה המתבקשת כרגע היא אם כל המטרה של עיבוד המילים הוא להזין אותם כקלט לרשתות נוירונים איך ב-W2V אנחנו משתמשים ברשת נוירונים? לכן עלינו למצוא דרך לייצג את המילים הללו באופן מתמטי, כך שהרשת תוכל לעבד אותן. אחת הדרכים לעשות זאת היא להשתמש בOne-hot שתיארנו למעלה, אמנם יש לייצוג זה את החסרונות שלו, אולם כאשר נקבל את הפלט של רשת הנוירונים נקבל עבור כל מילה ייצוג עם משמעות להקשר. המודל לוקח קורפוס של טקסט ויוצר ע"י one hot ווקטור לכל מילה. הווקטורים האלה מוכנסים כקלט אל המודל שהוא כאמור רשת נוירונים עם שכבה אחת. ב- word2vec משתמשים בייצוג מבוזר של מילה. המודל לוקח וקטור עם כמה מאות ממדים (נניח 1000). כל מילה מיוצגת על ידי חלוקת משקולות על פני אותם אלמנטים. אז במקום מיפוי של אחד לאחד בין אלמנט בווקטור למילה, הייצוג של מילה מתפרס על פני כל האלמנטים בווקטור, וכל אלמנט בווקטור תורם להגדרת מילים רבות. אם אני מתייג את הממדים בווקטור מילים היפותטי (אין תוויות כאלה שהוקצו מראש באלגוריתם כמובן), זה יראה בערך כמו ב**איור 48**. הווקטור שנוצר בסוף התהליך בא לייצג בצורה מופשטת כלשהי את 'המשמעות' של מילה. וכפי שנראה בהמשך, על ידי בחינת קורפוס גדול אפשר ללמוד וקטורי מילים המסוגלים לתפוס את מערכות היחסים בין מילים בצורה טובה להפליא. את הווקטורים שקיבלנו אנו הולכים להכניס אל רשת הנוירונים שלנו במחקר זה. מכיוון

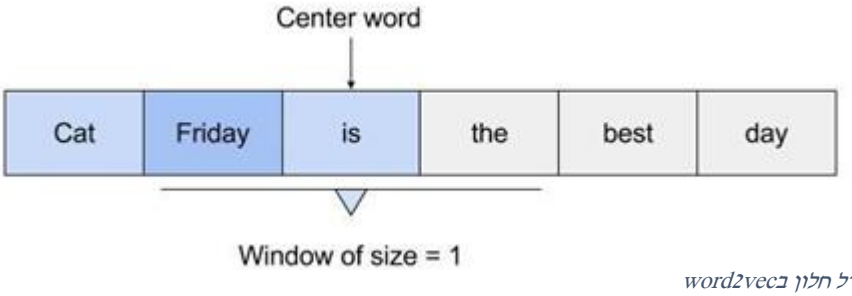


איור 49 מבנה רשת עיבוד word2vec



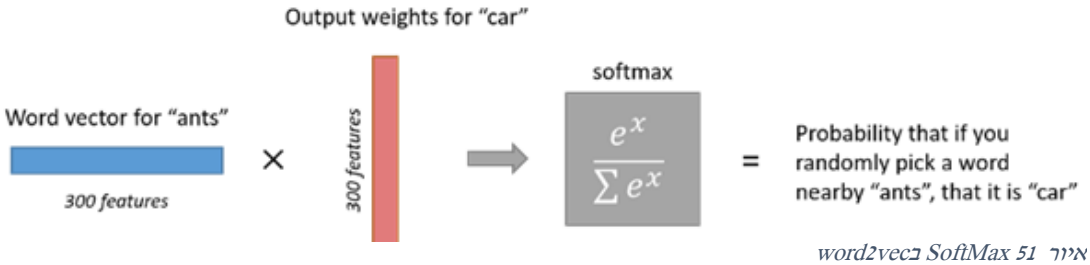
איור 50 פריסת רשת word2vec

שה'משמעות' של המילים נשמרת בשיטה זו אנו מקווים להגיע לתוצאות טובות יותר. עקב השימוש בשלב הראשון בווקטורים מסוג one -hot כפל של הווקטור במטריצת משקולות כלשהיא לדוגמא – W1 שווה ערך לבחירת שורה במטריצה W1 כמתואר ב**איור 49**. בתוך



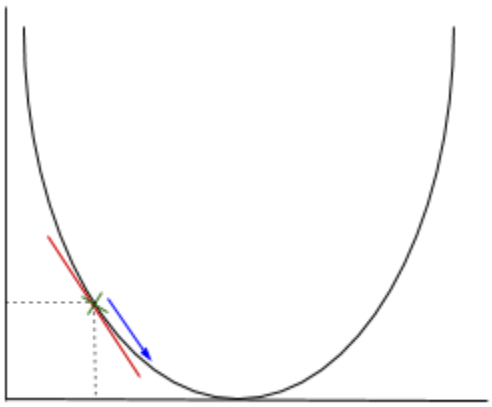
איור 52 גודל חלון בword2vec

הרשת מהשכבה הנסתרת לשכבת הפלט, ניתן להשתמש במטריצת משקל שנייה W2. כדי



איור 51 SoftMax בword2vec

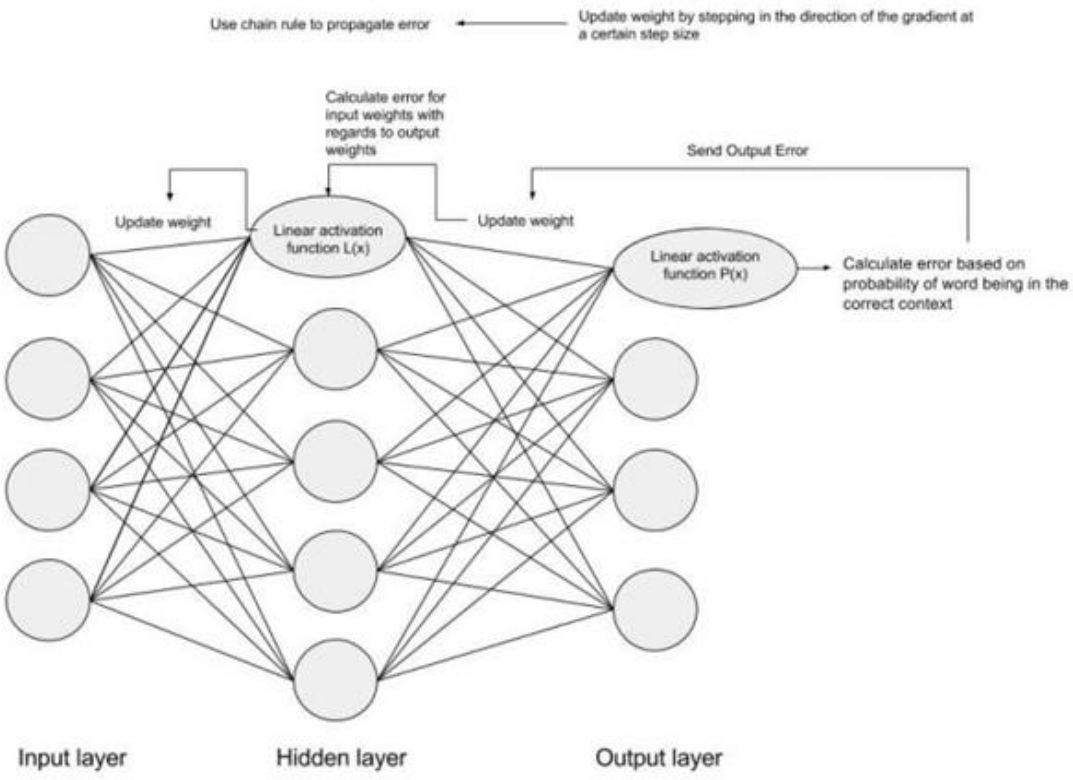
לחשב ניקוד לכל מילה באוצר המילים, נשתמש בפונקציית SoftMax לצורך השגת החלוקה הסופית של המילים. כמו שהסברנו לעיל דגם skip-gram הוא ההפך מדגם CBOW. הדגם בנוי עם מילת המיקוד כווקטור הקלט היחיד, ומילות היעד נמצאות כעת בשכבת הפלט. פונקציית ההפעלה של השכבה הנסתרת מסתכמת פשוט בהעתקת השורה המתאימה ממטריצת המשקולות W1 (העתקה ליניארית) כפי שראינו קודם. **באיור 50** ניתן לראות את ארכיטקטורת מודל הword2vec. במחקר שלנו אנו לומדים את וקטורי המילים עם **300** פיצ'רים. לפיכך השכבה הנסתרת עומדת להיות מיוצגת על ידי מטריצת משקל עם 10,000 שורות (אחת לכל מילה באוצר המילים שלנו) ו -300 עמודות (אחת לכל נוירון מוסתר). 300 פיצ'רים (זה גם מה שגוגל השתמשה במודל שפורסם ואומן על כמות גדולה של ידיעות חדשותיות מגוגל ניוז). מספר הפיצ'רים הוא "היפר פרמטר" שפשוט צריך לכוונן בהתאמה למודל הלומד, אם נסתכל בשורות של מטריצת המשקל הזו. המטרה הסופית של כל זה היא באמת רק לייצר מטריצת משקל בשכבה הנסתרת הזו – מה שיגרום לכך שנזין לשכבת הפלט



איור 53 stochastic gradient descent

את וקטור המילים 300 x 1 עבור המילה "נמלים" (לשם הדוגמא). שכבת הפלט היא מסווגת רגרסיה מסוג softmax. באופן ספציפי, לכל נוירון פלט יש וקטור משקל שהוא מכפיל את

עצמו מול ווקטור המילה מהשכבה הנסתרת, ואז הוא מיישם את הפונקציה exp (x) על התוצאה. לבסוף, בכדי לגרום לפלטים הסופיים להתכנס ל- 1, אנו מחלקים את התוצאה בסכום התוצאות מכל 10,000 צמתי הפלט. ב**איור 50** מופיעה המחשה לחישוב הפלט של נוירון אחד בשכבת הפלט עבור המילה "מכונית". אם לשתי מילים שונות יש "הקשרים" דומים מאוד (כלומר, אותן מילים עשויות להופיע סביבן), המודל שלנו צריך להפיק תוצאות דומות מאוד עבור שתי המילים הללו. אחת הדרכים של הרשת להוציא תחזיות הקשר דומות לשתי מילים כאלה היא כאשר וקטורי המילים דומים בערכים שלהם. כל ממד של הקלט עובר דרך כל צומת בשכבה הנסתרת. הממד מוכפל במשקל שמוביל אותו לשכבה הנסתרת. מכיוון שהקלט הוא וקטור ב-one hot, רק לאחד מצמתי הקלט יהיה ערך שאינו אפס (כלומר הערך של 1). משמעות הדבר היא שלכל מילה רק המשקולות שקשורות לצומת הקלט עם ערך 1



איור 54תהליך ה-backpropagation.

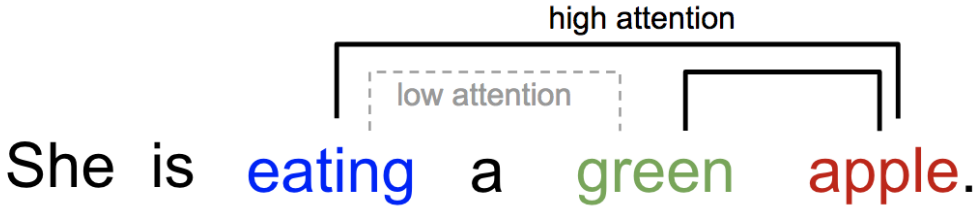
יופעלו. בהתחלה המשקולות שנמצאות במטריצה מאותחלות כערכים אקראיים. לאחר מכן מאמנים את הרשת על מנת להתאים את המשקולות לייצוג מילות הקלט. זה המקום בו שכבת הפלט הופכת חשובה. כעת, כאשר אנו נמצאים בשכבה הנסתרת עם ייצוג וקטורי של המילה, אנו זקוקים לדרך לקבוע כמה טוב ניבאנו שמילה תתאים להקשר מסוים. ההקשר של המילה הוא קבוצת מילים בתוך חלון סביבו כמוצג ב**איור 51**. מאיור 51 עולה כי ההקשר של המילה "Friday" כולל מילים כמו "cat" ו"is". מטרת הרשת העצבית היא לחזות ש"Friday" יופיע בהקשר הזה. אנו מפעילים את שכבת הפלט על ידי הכפלת הווקטור שעבר בשכבה הנסתרת (ששוות ערך למשקל של ווקטור ה-one hot * המשקל שנכנס לנוירון הפלט) עם ייצוג וקטורי של מילת ההקשר (שהיא הווקטור one - hot של מילת ההקשר כפול משקולות הנכנסות) . ב**איור 52** ניתן לראות הדמיה של מצב שכבת הפלט של מילת ההקשר הראשונה. הכפל לעיל

נעשה עבור כל צמד מילים בין מילה והערך ההקשרי שלה לפי מטריצת הערכים. לאחר מכן אנו מחשבים את ההסתברות שמילה שייכת לקבוצת מילות הקשר באמצעות הערכים הנובעים מהשכבה הנסתרת ומשכבת הפלט. לבסוף, אנו מיישמים ירידת שיפוע סטוכסטית (stochastic gradient descent) כמתואר **באיור 53** כדי לשנות את ערכי המשקולות על מנת לקבל ערך רצוי יותר עבור ההסתברות המחושבת. המשקל ישונה על ידי ביצוע צעד בכיוון הנקודה האופטימלית (באיור 54, הנקודה הנמוכה ביותר בתרשים – כמובן שהדרך היא בנגזרות ושימוש בכל השרשרת). הערך החדש מחושב על ידי 1.חיסור ערך המשקל הנוכחי את ערך הנגזרת בנקודה שבה המשקל מוגדל בקצב הלמידה (learning rate). השלב הבא הוא שימוש ב- Backpropagation, כדי להתאים את המשקולות בין שכבות מרובות. השגיאה המחושבת בסוף שכבת הפלט מועברת חזרה משכבת הפלט לשכבה הנסתרת על ידי החלת כלל השרשרת. Gradient descent משמשת לעדכון המשקולות בין שתי שכבות אלה. לאחר מכן השגיאה מותאמת בכל שכבה ונשלחת חזרה הלאה. ב**איור 54** ניתן לראות תרשים המייצג את תהליך ה-backpropagation.

4.3 Attention mechanism

איור 55 עמיר פרץ

תשומת לב חזותית אנושית מאפשרת לנו להתמקד באזור מסוים בעל "חשיבות גבוהה" (לדוגמא להסתכל על השערות הלבנות בשפם של עמיר פרץ – **איור 55**) תוך כדי תפיסת הדימוי שמסביב



איור 56 קשר בין מילים במשפט

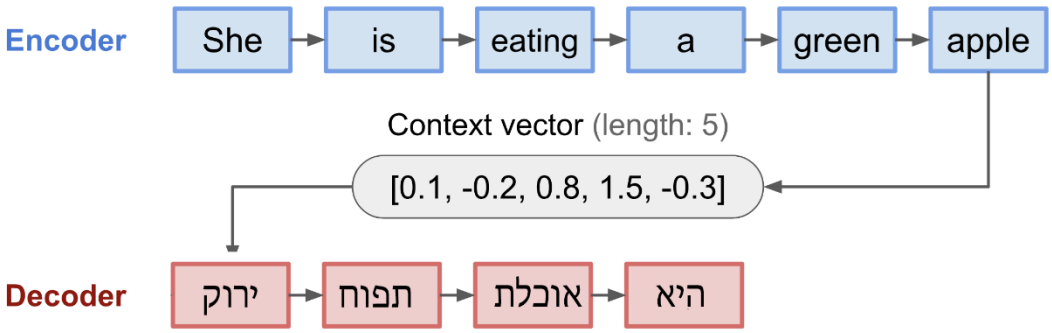
ב"רזולוציה נמוכה " (לדוגמא, מה דעתכם על הרקע של ארון הספרים ב**איור 55**) ואז להתמקד

בנקודה מסוימת בתמונה ולהסיק ממנה מסקנות בהתאם. לדוגמא בהינתן תיקון קטן של תמונה, הפיקסלים סביב הפגם יספקו לנו רמזים משמעותיים למה צריך להיות מוצג שם. ניתן לומר את זה כך - אנו מצפים לראות גבות עבות מעל העיניים של עמיר פרץ מכיוון שראינו את השפם המפורסם. עם זאת, דגל ישראל ברקע לא בהכרח יעזור לנו במקרה הזה. באופן דומה, אנו יכולים להסביר את הקשר בין מילים במשפט אחד או בהקשר מספיק קרוב אחר. **באיור 56** כאשר אנו רואים "eating", אנו מצפים להיתקל במילת של אוכל בקרוב מאוד. מונח הצבע ("green") מתאר את האוכל, אך ככל הנראה לא כל כך קשור למילה "eating" ישירות לעומת המילה "apple".

לפני שנדבר בפירוט על Attention נסביר כמה מושגים המאפשרים להשתמש במכניזם הזה.

4.3.1 מודל ה-seq2seq

מודל ה-seq2seq נולד בתחום מידול השפה[15]. באופן כללי, המטרה היא להפוך רצף קלט (מקור) לרצף חדש (יעד) ושני הרצפים יכולים להיות באורך שונה. דוגמאות למשימות טרנספורמציה כוללות תרגום מכונה בין מספר שפות בטקסט או באודיו, יצירת שיח עם תשובות לשאלות, או אפילו ניתוח משפטים לעצי דקדוק. למודל seq2seq בדרך כלל ארכיטקטורת מפענח (encoder) מקודד (decoder) , המורכבת מ :



encoder - decoder 57איור

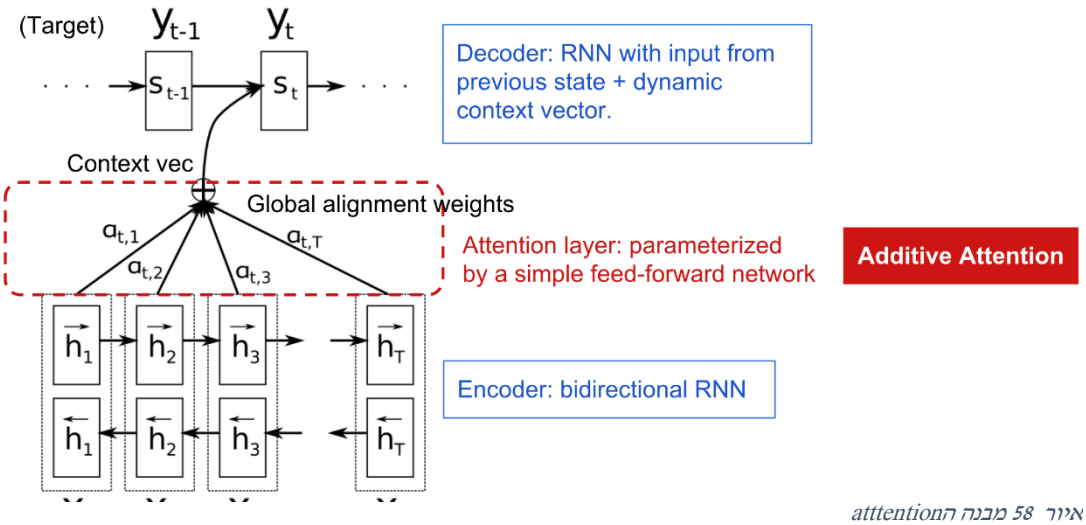
- **מקודד- Encoder** מעבד את רצף הקלט ודוחס את המידע לווקטור הקשר "**context vector**" כפי שניתן לראות **באיור 57** (המכונה גם sentence embedding או "וקטור מחשבה "thought") באורך קבוע. ייצוג זה הוא בעצם סיכום טוב של משמעות רצף המקור כולו, כלומר לקחנו משפט שלם ועיבדנו את כל המשמעות שלו לייצוג של ווקטור בגודל קבוע.

[15] Sequence to Sequence Learning with Neural Networks Ilya Sutskever, Oriol Vinyals, Quoc V. Le (Submitted on 10 Sep 2014 (v1), last revised 14 Dec 2014 (this version, v3))

- **מפענח-Decoder** מאותחל עם וקטור ההקשר כדי להוציא את הפלט המתאים. משתמשים רק במצב (state) האחרון של המקודד כמצב ההתחלתי (initial state) של המפענח.

הן המקודד והן המפענח הם רשתות נוירונים חוזרות (RNN). לדוגמא ניתן לממש אותם באמצעות יחידות LSTM או GRU.

הסיבה להולדת מנגנון הAttention, נובע מהעובדה הפשוטה - במשפטים ארוכים מאוד עם התקדמות הלמידה אנו מתחילים לשכוח את המילים הראשונות שנלמדו. לשם כך מממש מנגנון זו מכניקה של לקיחת כל הקלט כווקטור אחד ושילוב שלו באמצע תהליך הלמידה. כך אנו זוכרים את המילים הראשונות גם בסיום הלמידה[16].



איור 58 מבנה הattention

בפועל מנגנון ה- Attention נולד כדי לעזור לשנן משפטים מקוריים ארוכים במשימה של תרגום מכונה נוירוני (NMT-neural machine translation). במקום לבנות וקטור קונטקסט יחיד מתוך המצב הנסתר האחרון של המקודד, ה'שיקוי' הסודי שהומצא על ידי ה-Attention הוא ליצור קיצורי דרך בין וקטור הקונטקסט לכניסת קלט המקור כולו. המשקלים של קיצורי דרך אלה ניתנים להתאמה אישית עבור כל אלמנט פלט כמתואר **באיור 58**. לכן בזמן שלווקטור הקונטקסט יש גישה לכל רצף הקלט, איננו צריכים לדאוג מן השכחה, שהרי תמיד יש לו גישה אל המקור. היישור (alignment) בין המקור למטרה נלמד ונשלט על ידי וקטור הקונטקסט. למעשה, וקטור הקונטקסט צורך שלוש פיסות מידע:

1. מקודד המצבים הנסתרים של המקודד;
2. המצבים הנסתרים של המפענח;
3. היישור (alignment) בין המקור והמטרה.

[16] Neural Machine Translation by Jointly Learning to Align and Translate Dzmitry Bahdanau, Kyunghyun Cho, Yoshua Bengio (Submitted on 1 Sep 2014 (v1), last revised 19 May 2016 (this version, v7))

נסביר את המנגנון בצורה פשוטה ולאחר מכן נציג את הנוסחאות המתמטיות.

בשלב הראשון אנו מכניסים את המשפטים אל המקודד, המקודד מבצע הרצה על גבי הקלטים, המקודד ברשתות חוזרות מוציא תמיד שני פלטים:

1. ווקטור הפלט של הרשת
2. וקטור השכבות החבויות.

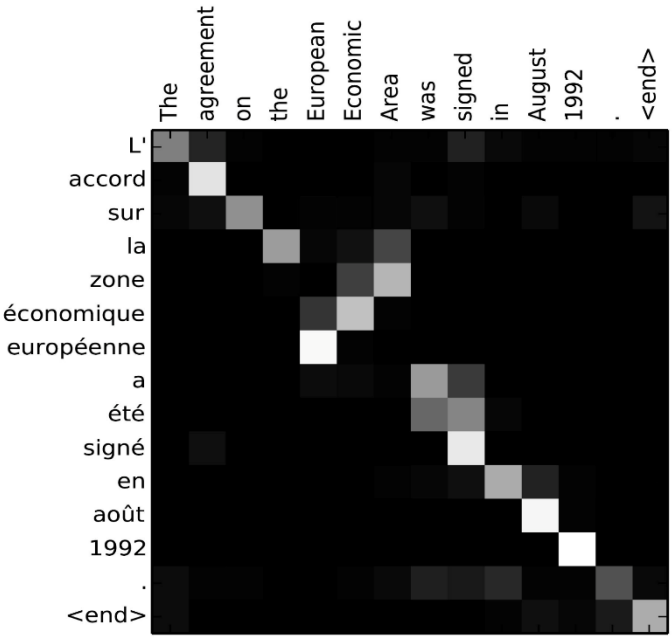
בשלב השני אנו מכניסים את הווקטורים הללו כקלט קל פונקציית הAttention, פונקציה זו מבצעת לכל מילה "ישור" או Allignment מה שנקרא, פונקציית היישור. מטרת פונקציה הזו היא לחשב את הקשר בין המילה לחלק בווקטור המצב הנסתר (שמייצג את כל המשפט).

כעת נגדיר את המנגנון שהוסבר בצורה מתמטית. נניח, יש לנו רצף מקור **x** באורך n וננסה להוציא רצף יעד **y** באורך m:

$$\boldsymbol{x} = [x_1, x_2, \ldots, x_n]$$

$$\boldsymbol{y} = [y_1, y_2, \ldots, y_m]$$

(משתנים מודגשים מציינים שהם וקטורים.)



איור 59 תוצאות הattention

המקודד הוא רשת חוזרת דו כיוונית (bidirectional) עם מצב נסתר קדימה (forward hidden state) $\overrightarrow{\boldsymbol{h}_i}$ וגם אחורה (backward hidden state) $\overleftarrow{\boldsymbol{h}_i}$. שרשור פשוט (concatenation) של שני

הווקטורים מייצג את מצב המקודד. המוטיבציה היא לכלול את המילים שקדמו בהערה של מילה אחת.

$$\boldsymbol{h}_i = [\overrightarrow{\boldsymbol{h}_i}^T ; \overleftarrow{\boldsymbol{h}_i}^T]^T , i = 1, \ldots, n$$

ברשת המפענח יש מצב נסתר $st = f(\boldsymbol{s}_{t-2}, y_{t-1}, \boldsymbol{c}_t)$ עבור מילת הפלט במיקום t, t = 1, ..., m, כאשר וקטור ההקשר $\boldsymbol{c}_t$ הוא סכום של המצבים הנסתרים של רצף הקלט, משוקלל על ידי ניקוד יישור (alignment scores) לפי מודל היישור:

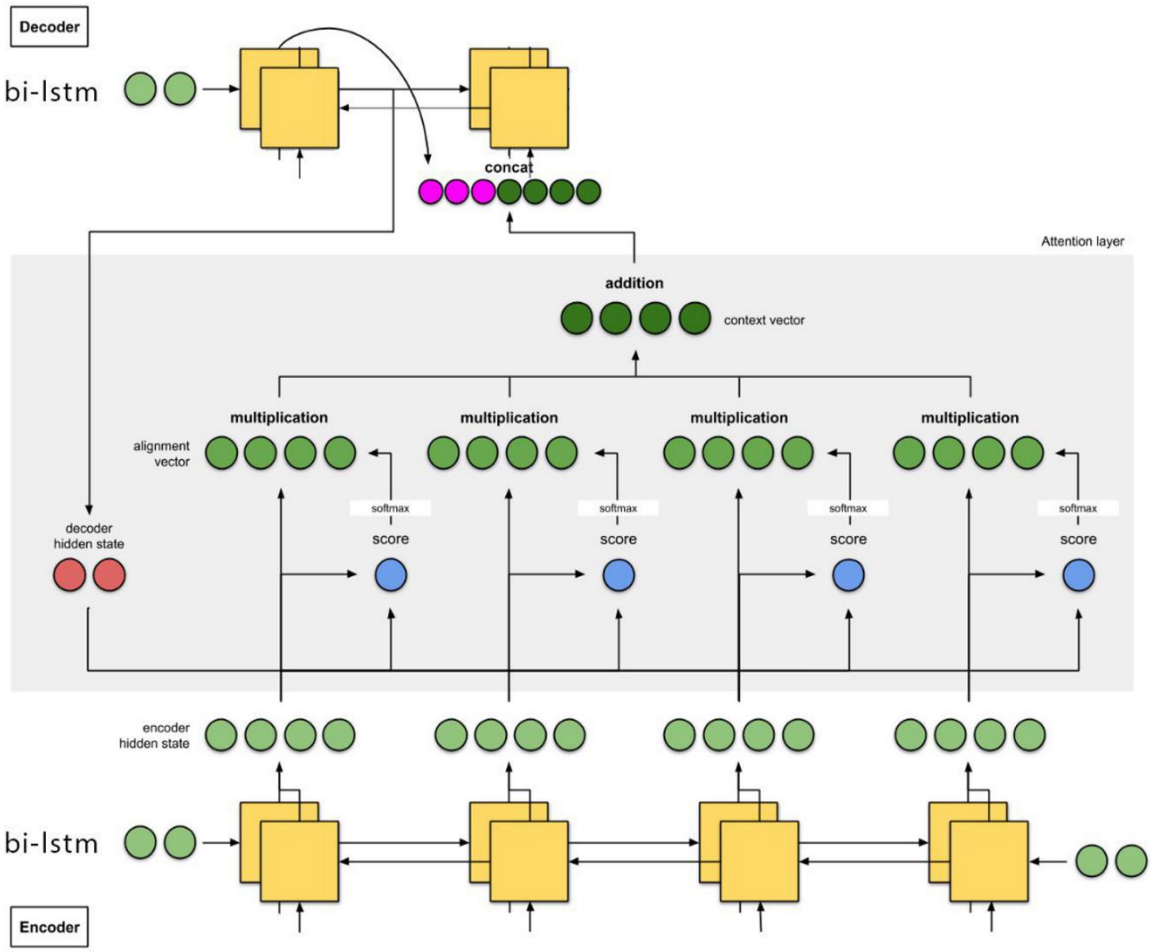
$$\begin{aligned}
\mathbf{c}_t &= \sum_{i=1}^{n} \alpha_{t,i} \boldsymbol{h}_i && \text{; Context vector for output } y_t \\
\alpha_{t,i} &= \text{align}(y_t, x_i) && \text{; How well two words } y_t \text{ and } x_i \text{ are aligned.} \\
&= \frac{\exp(\text{score}(\boldsymbol{s}_{t-1}, \boldsymbol{h}_i))}{\sum_{i'=1}^{n} \exp(\text{score}(\boldsymbol{s}_{t-1}, \boldsymbol{h}_{i'}))} && \text{; Softmax of some predefined alignment score..}
\end{aligned}$$

מודל היישור מקצה ניקוד α_t, i לזוג הערכים - הקלט במיקום ה-i והפלט במיקום t, (y_t, x_i) בהתבסס על מידת התאמתם. הסט $\{\alpha_t, i\}$ הם משקלים המגדירים כמה מכל מצב נסתר צריך להילקח בחשבון עבור כל פלט. במחקר שלנו השתמשנו ב**מודל היישור של Bahdanau**, במודל זה הניקוד ליישור α מוגדר על ידי רשת קדימה (feed-forward network) עם שכבה נסתרת אחת והרשת הזו מאומנת יחד עם חלקים אחרים של המודל. פונקציית הניקוד היא אפוא בצורה הבאה, בהתחשב בעובדה ש $tanh$ משמשת כפונקציית ההפעלה לא לינארית:

$$\text{score}(\boldsymbol{s}_t, \boldsymbol{h}_i) = \mathbf{v}_a^\top \tanh(\mathbf{W}_a[\boldsymbol{s}_t; \boldsymbol{h}_i])$$

הסיבה לשימוש בפונקציית האקטיבציה tanh לחישוב הscore ספציפית במקרה הזה, מכיוון ששיטה זו מממשת רשת קדימה, (feed-forward network) ובד"כ במשימות של קלסיפיקציה בין שני מחלקות ורשתות קדימה משתמשי בtanh. לאחר חישוב הScore נבצע SoftMax ע"מ לקבל תוצאות מותאומות בטווח של 0 עד 1. **באיור 59** מוצגת מטריצת הניקוד ליישור. מטריצה זו היא תוצר לוואי נחמד שמראה במפורש את ההתאמה בין מילות המקור ומילות היעד ובכך מסכם את עבודת המודל בצורה חזותית. ניתן לומר לסיכום כי כלי זה הוכיח את עצמו במספר רב של משימות, והוא נושא עיקרי במחקר שלנו.

5 מבנה המודל שיטת העבודה ותוצאותיה



5.1 כלים

- Python - שפת תכנות דינמית מהנפוצות המבוקשות והפופולאריות ביותר פייתון היא שפה מרובת שיטות אשר קיימת עבורה ספרייה סטנדרטית גדולה וענפה של הרחבות וכלים. שפה זו שימשה עבורנו כפלטפורמה, לשימוש בכל הכלים אשר נצרכנו עבורם ועזרה לנו להוציא לפועל ולממש את הפרויקט. יש לציין כי פייתון מצטיינת בתמיכה הרבה שלה עבור מניפולציות על טקסט בקלות ומהירות.

- Gensim– ספריית קוד פתוח המשמשת לניתוח טקסטים ועיבוד מילים לווקטורים תוך מתן כלים לשימוש בווקטורים ובייצוגם. כלי זה שימש עבורנו ליצירת מודלי ווקטורים עבור הטקסטים השונים אשר עבדנו איתם ותרם לחוויית תכנות וניתוח קלות יותר.

- TensorFlow - ספריית קוד פתוח ללמידת מכונה, המפותחת על ידי חברת גוגל לבנייה ואימון רשתות עצביות. כלי זה שימש עבורנו לצורך בניית רשתות הנוירונים השונות שיצרנו לכל אורך הפרויקט.

- Keras - ספריית קוד פתוח, אשר רצה על גבי TensorFlow (ועוד ספריות אחרות) כלי זה תוכנן כדי לאפשר ניסויים מהירים עם רשתות עצביות עמוקות, והוא מתמקד בהיותו ידידותי למשתמש, מודולרי וניתן להרחבה. כלי זה שימש עבורנו פלטפורמה נוחה וידידותית למימוש ואימון הרשתות.

- Pandas – ספריה עוצמתית לעבודה עם קבצי csv נוחים וקלים לשליפה של אינדקסים מתוך טבלה גדולה.

- sk-learn – ספריה נפוצה הממשת קבוצה רחבה של אלגוריתמים ללמידת מכונה ולביצוע סטטיסטיקות.

5.2 סביבת העבודה

את קוד הפייתון הרצנו דרך פלטפורמה נפוצה מאד בתחום הData science, הנקראת **Jupyter**, היתרון בסביבה זו היא הקלות בה אפשר להריץ קוד בצורה נפרדת, והיכולת לשתף את הקוד ולהטמיע הסברים ותמונות לצד הקוד. קישור למחברת הקוד מופיע בנספח 2, התקנת סביבת העבודה מופיעה בנספח 1.

5.3 חומרה

את המודלים הרצנו על מחשב נייד אישי, על כרטיס גרפי מסוג NVIDIA GTX 1060, עם אופטימיזציות ברמת התוכנה של ספריית CudaNN.

5.4 קורפוס הקלט

הקלט הגיע מ50 אלף משפטים בעברית, SVLM Hebrew Wikipedia Corpus מתוך הקורפוס של ד"ר ורד זילבר ופרופסור עמי מויאל כחלק מפרויקט של בדיקת שכיחות פונמות בשפה העברית. המשפטים במקור הגיעו מתוך ויקיפדיה העברית.

דוגמא למשפט מתוך הקלט: "פרסומים עיקריים מקורות המחשבה הצבאית המודרנית משרד הביטחון ההוצאה לאור קישורים חיצוניים אתר האינטרנט של המרכז הירושלמי לענייני ציבור ומדינה."

5.5 תיוג המשפטים

השתמשנו ב -Hebrew Dependency Parser - תייגן של ד"ר יואב גולדברג המוצא קשרים במשפט כמו נושא נשוא במשפטים בעברית. לדוגמא עבור הקלט הבא:

```
קורות חייו צמרת למד בחוגים לחינוך ופילוסופיה באוניברסיטה ה
עברית בירושלים
```

התייגן מוציא פלט בצורה הבאה:

1	קורות	_	NNT	NNT	F|P	4	SBJ	_	_		
2	חיין	_	NN_S_PP	NN_S_PP	M|P|suf_M|suf_S|suf_3	1	dep	_	_		
3	צמרת	_	NNT	NNT	F|S	2	dep	_	_		
4	למד	_	VB	VB	M|S|3|PAST|PAAL	0	ROOT	_	_		
5	ב	_	PREPOSITION	PREPOSITION	_	4	dep	_	_		
6	חוגים	_	NN	NN	M|P	5	dep	_	_		
7	ל	_	PREPOSITION	PREPOSITION	_	6	dep	_	_		
8	חינוך	_	NN	NN	M|S	9	CONJ	_	_		
9	ו	_	CONJ	CONJ	_	7	dep	_	_		
10	פילוסופיה	_	NN	NN	F|S	9	CONJ	_	_		
11	ב	_	PREPOSITION	PREPOSITION	_	9	dep	_	_		
12	אוניברסיטה	_	NN	NN	F|S	11	dep	_	_		
13	ה	_	DEF	DEF	_	14	dep	_	_		
14	עברית	_	JJ	JJ	F|S	12	ADJ	_	_		
15	ב	_	PREPOSITION	PREPOSITION	_	4	dep	_	_		
16	ירושלים	_	NNP	NNP	_	15	dep	_	_		

כאשר אנחנו שלפנו את כל המשפטים בעלי נושא מובהק. סה"כ כ35 אלף משפטים.

5.6 חלוקת המשפטים

מתוך 50 אלף המשפטים שלפנו רק משפטים בעלי נושא מובהק והגענו לכ35 אלף משפטים. ביצענו חלוקה ל2 קבוצות של משפטים, קבוצה אחת שהיא שלושת רבעי מהמשפטים ששימשה אותנו לתהליך הלמידה ושאר המשפטים השארנו בכדי לשלול Over Fitting .

5.7 עיבוד מקדים Word Embedding

על המשפטים ביצענו עיבוד מקדים בשיטת Word2vec, בנינו מודל מן המשפטים עם הארגומנטים הבאים :

- **min_count = 1:** התדירות המינימלית. המודל יתעלם ממילים שאינן מספקות את הסכום המינימלי. מילים נדירות במיוחד אינן חשובות בדרך כלל, אז הכי טוב להיפטר מאותן. אם מערך הנתונים ממש זעיר, ואז זה לא ממש משפיע על הדגם.

- **window=5 :** המרחק המרבי בין מילת היעד למילה הסמוכה. אם מיקום המילה השכנה גדול מרוחב החלון המרבי משמאל לימין, אז חלק מהשכנים אינם נחשבים כקשורים למילת היעד. בתיאוריה, חלון קטן יותר אמור לתת מונחים קשורים יותר.

- **iter=100:** מספר האיטרציות לעיבוד המילים.

- **EMBEDDING_DIM = 300** : גודל הווקטור לכל מילה.

- **Batch size=64**

- **אורך משפט מקסימלי = 10.**

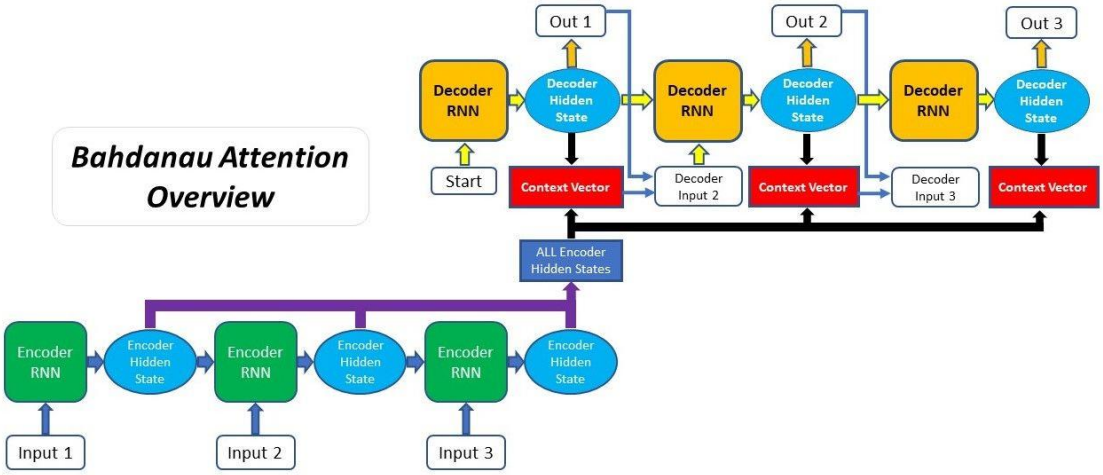
5.8 שכבת המקודד

שכבת המקודד מומשה בהתאם לסוג כל רשת – LSTM ,bi-LSTM,GRU. עם הקלטים הבאים:
enc_units=1024, return_sequences=True, return_state=True.

- `return_sequences`: Boolean. Whether to return the last output. in the output sequence, or the full sequence.
- `return_state`: Boolean. Whether to return the last state in addition to the output.

5.9 שכבת הAttention

בהתאם למוסבר בפרקי המחקר התיאורטי, ברשתות שמימשנו, השתמשנו בשיטת ה - Attention של בהדאנהו –

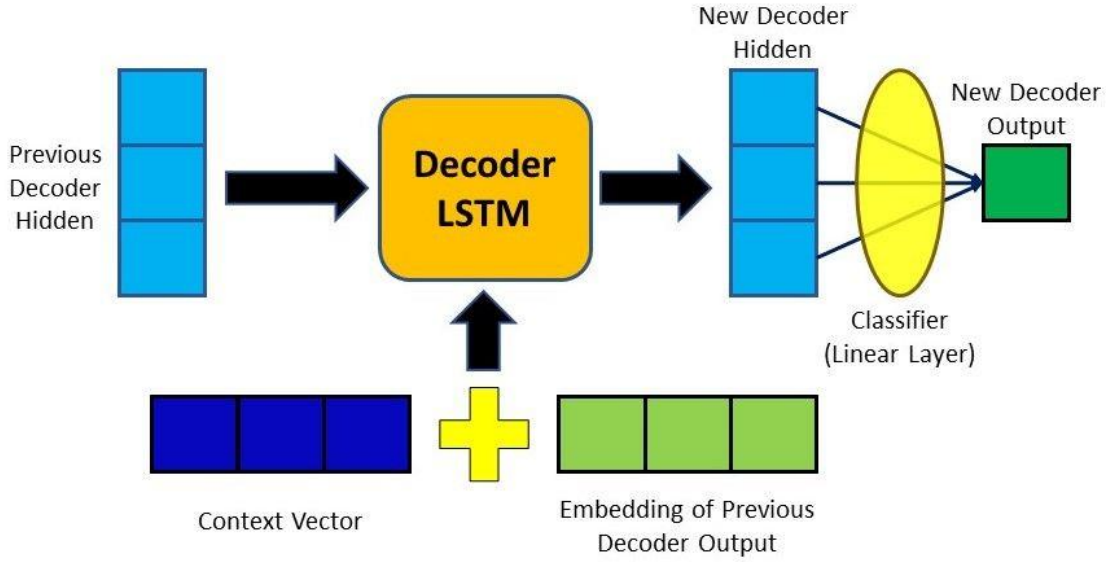


בשיטה זו מתבצעים שלבי העיבוד הבאים:

1. שלב הקידוד - המקודד מייצר מצבים נסתרים של כל רכיב ברצף הקלט.
2. Alignment Scores - חישוב ציוני יישור בין המצב הנסתר הקודם לפענוח לבין כל אחד מהמצבים הנסתרים של המקודד (הערה: ניתן להשתמש במצב החבוי האחרון של המקודד כמצב מוסתר ראשון במפענח)
3. ביצוע Softmax על ציוני היישור - ציוני היישור עבור כל מצב נסתר מקודד משולבים ומוצגים בווקטור יחיד עליהם מתבצעת ונקציית הSoftmax.
4. חישוב וקטור ההקשר Calculating the Context Vector- המצבים הסמויים המקודדים וציוני היישור שלהם בהתאמה מוכפלים ליצירת וקטור ההקשר
5. פענוח הפלט - וקטור ההקשר משויך לפלט המפענח הקודם ומוזן לתוך RNN המפענח באותו שלב זמן יחד עם המצב הנסתר הקודם של המפענח בכדי לייצר פלט חדש.
6. התהליך (שלבים 2-5) חוזר על עצמו בכל שלב בפענוח.

5.10 שכבת המפענח

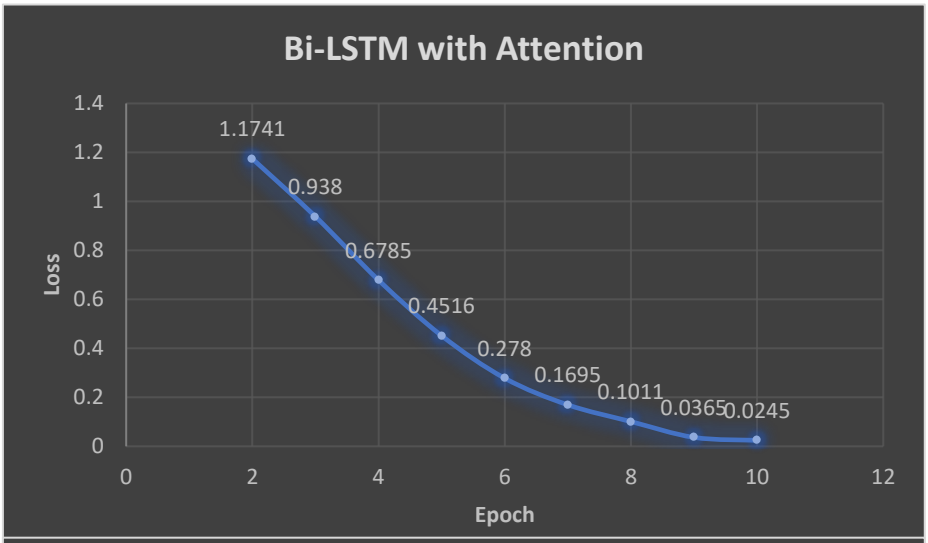
וקטור ההקשר שייצרנו לאחר מכן יחובר לפלט המפענח הקודם. כעת הוא מוזן לתא RNN המפענח כדי לייצר מצב נסתר חדש והתהליך חוזר על עצמו משלב 2. הפלט הסופי לשלב הזמן מתקבל על ידי העברת המצב הנסתר החדש דרך שכבה לינארית, המשמשת כמסווג ע״י ציוני ההסתברות של המילה החזויה הבאה.

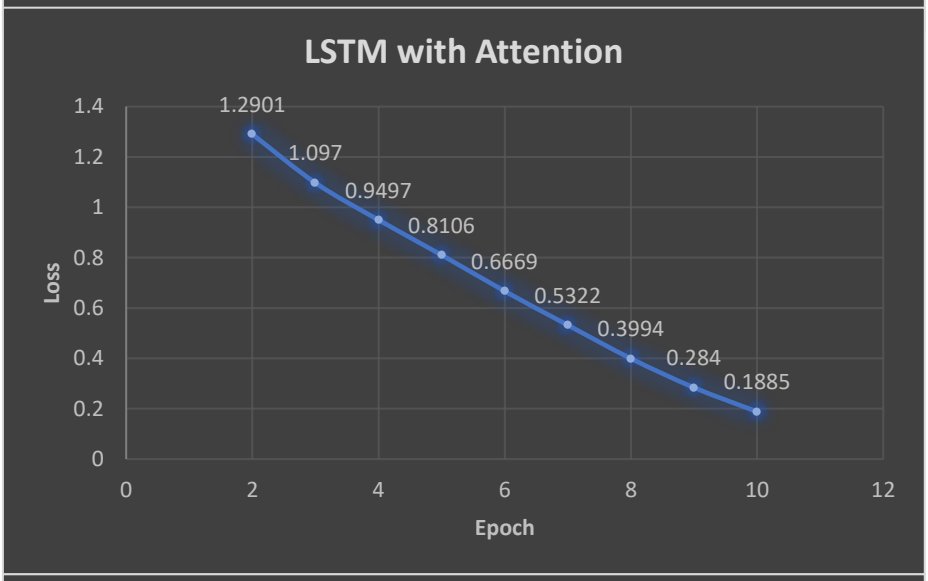


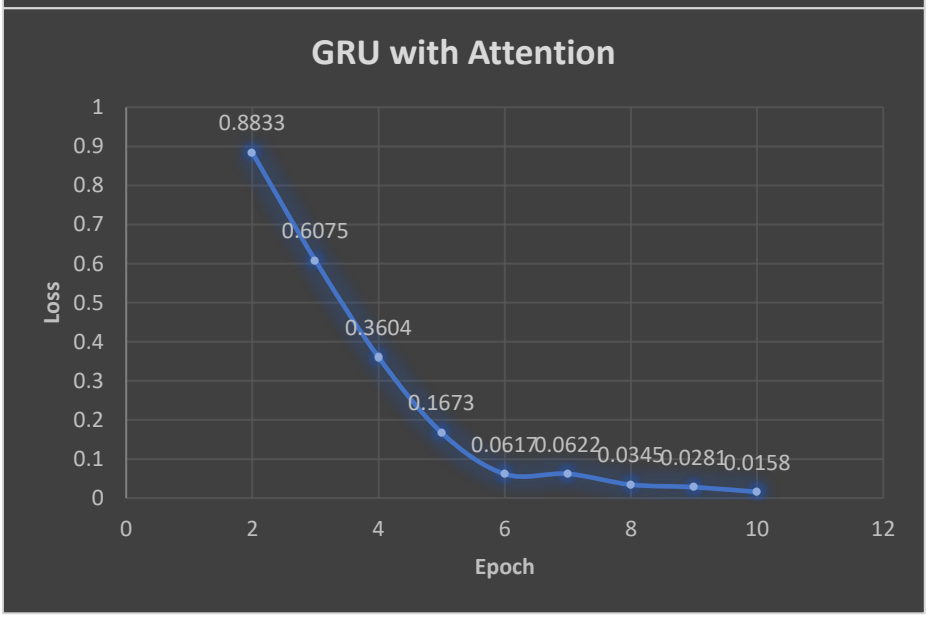
וקטור הקשר ויציאה קודמת יהיו הקלטים ליצירת מצב נסתר חדש של המפענח.

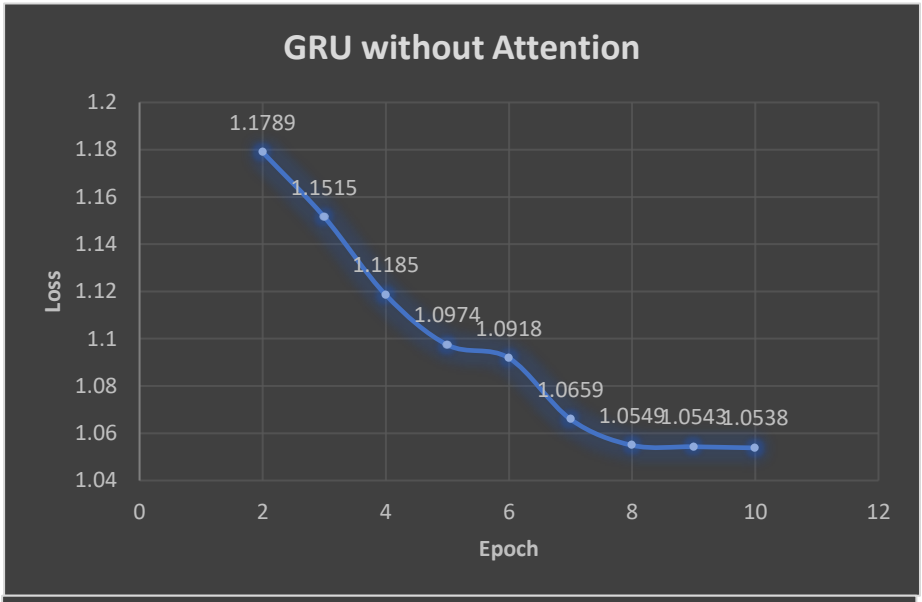
שלבים 2 עד 4 חוזרים על עצמם עד שהמפענח מייצר אסימון סוף משפט (end of file- EOF) או שאורך הפלט עולה על האורך המרבי האפשרי.

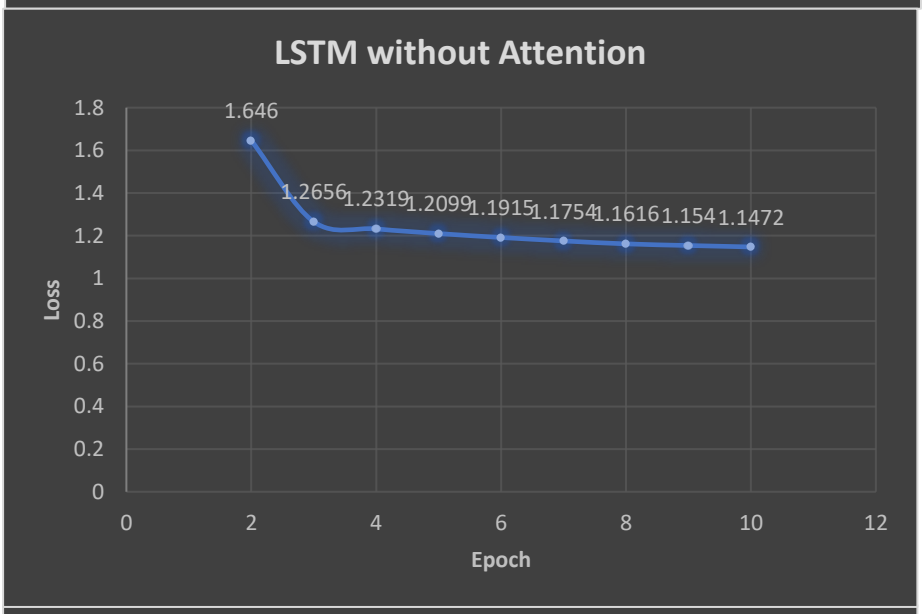
5.11 גרפים של תוצאות

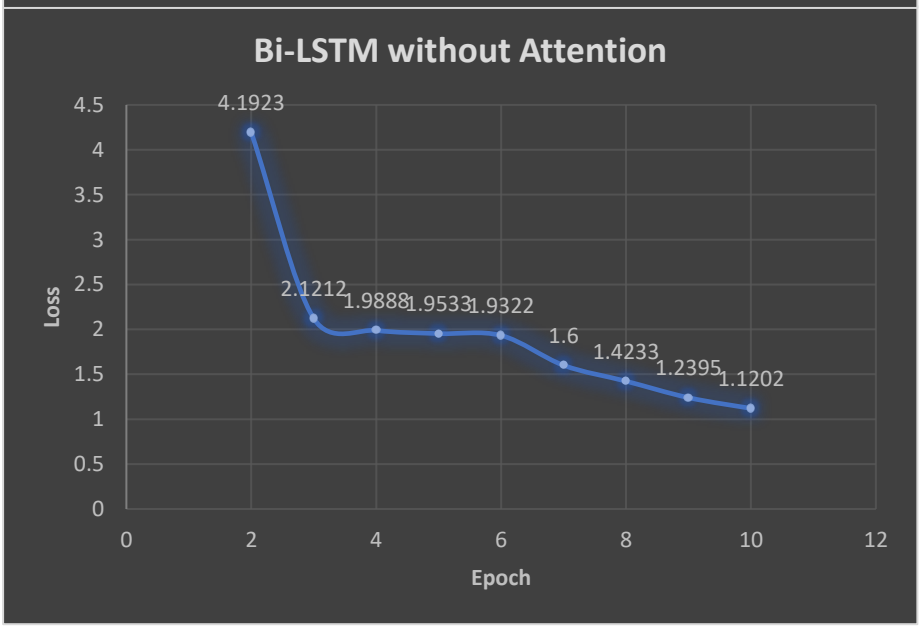







GRU without Attention
Loss
Epoch
1.2
1.18
1.16
1.14
1.12
1.1
1.08
1.06
1.04
0
2
4
6
8
10
12
1.1789
1.1515
1.1185
1.0974
1.0918
1.0659
1.0549
1.0543
1.0538


LSTM without Attention
Loss
Epoch
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
0
2
4
6
8
10
12
1.646
1.2656
1.2319
1.2099
1.1915
1.1754
1.1616
1.154
1.1472


Bi-LSTM without Attention
Loss
Epoch
4.5
4
3.5
3
2.5
2
1.5
1
0.5
0
0
2
4
6
8
10
12
4.1923
2.1212
1.9888
1.9533
1.9322
1.6
1.4233
1.2395
1.1202

5.12 זמני ריצה

- עבור רשת מסוג bi-LSTM עם Attention: זמן ממוצע להרצת epoch אחד: 138 שניות.
- עבור רשת מסוג LSTM עם Attention: זמן ממוצע להרצת epoch אחד: 95 שניות
- עבור רשת מסוג GRU עם Attention: זמן ממוצע להרצת epoch אחד: 88 שניות.
- עבור רשת מסוג bi-LSTM ללא Attention: זמן ממוצע להרצת epoch אחד: 112 שניות.
- עבור רשת מסוג LSTM ללא Attention: זמן ממוצע להרצת epoch אחד: 83 שניות
- עבור רשת מסוג GRU ללא Attention: זמן ממוצע להרצת epoch אחד: 79 שניות.

5.13 בדיקת דיוק למטרת שלילת 'התאמת יתר' (overfitting)

התאמת יתר (overfitting) היא בעיה יסודית בסטטיסטיקה ובלמידת מכונה שבה המודל מותאם יתר על המידה לאוסף מסוים של נתונים (למשל האוסף שהיה זמין לשם אימונו) ועל כן מצליח פחות בבצוע תחזיות. התאמת יתר מתרחשת כאשר המודל נקבע על ידי יותר פרמטרים מאשר הנתונים מצדיקים. עודף הפרמטרים מאפשר למודל ללמוד את הרעש הסטטיסטי כאילו הוא מייצג התנהגות אמיתית. בכדי לשלול אפשרות זו נהוג לבדוק באופן ישיר את יכולת המודל לבצע תחזיות על נתונים חדשים שלא שימשו בעת אימונו המקורי דרך זאת נקראת אימות צולב או cross validation.

במחקר זה בדיקת **רמת הדיוק Accuracy** קצת מורכבת מסיבה עיקרית אחת - מבחן הדיוק בשיטה מקובלת בוחן עבור כל תחזית עד כמה "קרובה" באחוזים למילה הרצויה. אולם במחקר שלנו, יש לחקור כיצד להגדיר את מדד ה'קירבה' בין שתי המילים. האם נגדיר קירבה ע"פ דמיון Cosine similarity בין הווקטורים? או נבחן דיוק ע"פ אחוז המשפטים בהם המודל הצליח לחזות, מתוך כל הtest – set. מצד אחד לא יהיה נכון לבחון ע"פ אחוז הצלחה מתוך כלל המשפטים, כי אין לנו שום אינדיקציה למידת הקירבה האמיתי של המודל בין המילים. וכן לא בהכרח נכון למדוד את הקירבה בעזרת מרחק וקטורי, כי אולי בחינת מרחק טכני בין המילים במשפט (המילה המתקבלת לעמת המילה הרצויה) היא הנכונה נציג דוגמאות מתאימות בהמשך.

לפיכך ביצענו שלושה מבחנים:

1. בחינת אחוזי הצלחה מתוך כלל המשפטים: לקחנו 8750 משפטים שהם כאמור 25% מהמשפטים (משפטים אלו לא נלמדו מעולם ע"י המודל), ובדקנו בכמה מן המשפטים המודל הצליח לחזות את המילה הרצויה.
2. בחינת קירבה בין וקטורים, השתמשנו בנוסחה ידועה למרחק וקטורים הנקראת Cosine similarity ע"פ הנוסחה הבאה כאשר A וB הן הווקטורים של המילים:

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}},$$

3. בדיקת מרחק בתוך המשפט, בין המילה המתקבלת לבין מילת הנושא. לדוגמה: "ילד כותב בתוך דלי" המרחק בין המילה – ילד, למילה דלי הוא 3. את המרחק חילקתי בגודל המילה פחות אחד.

$$\frac{i_A - i_B}{length(\text{sequence}) - 1}$$

תוצאות מבחני הדיוק הנ"ל עבור 8000 משפטים:

מס' מבחן	דיוק
1	18%
2	21%
3	51%

6 דיון ומסקנות

מבחינת זמני הריצה: כפי שניתן לראות, זמני ריצת מודלים הכוללים את מכניקת הattention ארוכים יותר ממודלים ללא מכניקה זו. כמו כן בזמן הריצה מודל הgru הוא הקצר ביותר ומודל הbi-lstm הוא הכי ארוך מבין השלושה.

מבחינת מדד הloss: כפי שניתן להסיק מן התוצאות, אפשר לראות כי לפי המדד של פונקציית הLoss שמומשה ע"י ה-tf.keras.losses.SparseCategoricalCrossentropy. עקומת הלמידה והשינוי עבור מודל ה**Bi-LSTM with Attention** הוא מביא את העקומה הטובה ביותר. אולם לפי התוצאה הטובה יותר עבור 10 אפוקים, ניתן לראות כי מודל ה**GRU** מגיע לתוצאות טובות יותר לבסוף. מבחינת המחקר שלנו בחנו את תהליך הלמידה מבחינת כל אחד מהמודלים לעיל. מודל ה**LSTM** הרגיל הביא תוצאות פחות טובות משאר המודלים. המודל העיקרי שחקרנו בפרויקט היה מודל הbi-LSTM, ונוכחנו לדעת כי מבחינת עקומת הלמידה הוא אכן הציג את התוצאות הטובות ביותר. כאמור ראינו שמבחינת ההשוואה רשת bidirectional מביאה תוצאות טובות בפער לעומת רשת קלאסית של LSTM. מכניקת הAttention גם היא השיגה תוצאות ראויות.

מבחינת מדד הדיוק: תוצאות המבחנים הפתיעו מאוד לרעה, הפער בין מדד הLoss לבין מדד הaccuracy לא היה מצופה. לשם כך בדקנו ידנית את הפער בין מלת הפלט לבין מילת המטרה וגילינו מספר הבדלים שלדעתנו הם הסיבות לתוצאות הנ"ל.

לדוגמה: עבור המשפט – *'עמוס עוז הסופר המוערך בישראל הגיע לבקר'* התייגן שלנו זיהה את 'עוז' כמילת הנושא לעומת התייגן של ד"ר יואב גולדברג שזיהה את 'עמוס' כנושא. כלומר יש להתייחס ולקחת בחשבון מקרים בהם אורך הנושא גדול ממילה אחת. כמובן מצאנו מספר רב של טעויות תיוג גם בתייגן של ד"ר יואב גולדברג.

המסקנות שלנו מבחינת תוצאות הדיוק היו, כי יש לבחון את המודל שוב עבור קלט איכותי של מילים מתויגות בצורה טובה יותר מהתייגן של ד"ר יואב גולדברג. כמו כן יש לבחון לעומק שיטה ייחודית לבדיקת רמת תיוג במקרים כמו במחקר שלנו.

7 מבט אל העתיד

כמחקר עתידי יש לבחון מספר גורמים מעבר לתוצאות לעיל:

1. אנו השתמשנו בword Embedding מסוג word2vec . יש לבחון את התוצאות מול שיטות אחרות כמו Fast Text Bert וכמובן Glove.
2. יש לבחון מקרים שונים במכניקת ה attention ישנן המון שיטות לבצע את המכניקה. אנו השתמשנו בשיטה של Bahdanau אולם יש לבחון גם בשיטה של Luong.
3. ראוי לבדוק את שילוב של encoder ו decoder מסוגים שונים. במחקר שלנו הן ה encoder והן הdecoder היו בנויים מאותו סוג רשת (bi-lstm,lstm,gru) כדאי אולי לבחון שילוב של רשתות שונות.
4. יש לבחון מודל חדש של Attention שמתמקד דווקא בשמות עצם או שמות פרטיים (כלומר לשלב NER) ע"פ רוב נושא הוא מהסוג הזה של חלקי הדיבר.
5. בחינת מודל עבור מקרים מורחבים כמו נשוא מורחב, מושא ועוד תפקידים תחביריים.
6. בדיקה של מכניקת ה Attention לסירוגין, וכך להקל על כובד וזמן הלמידה.
7. בחינת מבחן רמת דיוק ייחודי המתאים למשימות מהסוג של מחקר זה, ניתן לבחון בצורה של קבלת תיוג כווקטור בינארי עם הערך 1 במיקום מילת הנושא. לדוגמה: עבור המשפט "הילד הרשע קיבל מכה" המילה ילד באינדקס 0 היא מילת הנושא, לכן נייצג את המילה כווקטור באופן הבא: [0,0,0,1]. ובזה ניתן לעשות רדוקציה לבעיית הכרעה, שהינה בעיה קלה יותר בתחום למידת המכונה.
8. לבחון את המודלים על משפטים המתוייגים טוב יותר ובצורה איכותית יותר.
9. לבחון את המודלים על שפות זרות כדוגמת אנגלית שלה יש תייגנים טובים יותר וכן היא שפה פחות מסובכת (אינה עשירה לשונית כמו העברית).

8 ביבליוגרפיה

1. M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," in IEEE Transactions on Signal Processing, vol. 45, no. 11, pp. 2673-2681, Nov. 1997. doi: 10.1109/78.650093
2. Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." arXiv preprint arXiv: 1409.0473 (2014).
3. S. Hochreiter and J. Schmidhuber. Long Short-Term Memory. Neural Computation, 9(8): 1735–1780, November 1997
4. Alex Graves and Jürgen Schmidhuber. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. Neural Networks, 18(5): 602–610, July 2005
5. Reza Ghaeini, Sadid A. Hasan, Vivek V. Datla, Joey Liu, Kathy Lee, Ashequl Qadir, Yuan Ling, Aaditya Prakash, Xiaoli Z. Fern, and Oladimeji Farri. Dr-bilstm: Dependent reading bidirectional LSTM for natural language inference. CoRR, abs/1802.05577, 2018.
6. Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. CoRR, abs/1406.1078, 2014.
7. Neural Machine Translation by Jointly Learning to Align and Translate Dzmitry Bahdanau, Kyunghyun Cho, Yoshua Bengio (Submitted on 1 Sep 2014 (v1), last revised 19 May 2016).
8. https://www.digitaltrends.com/cool-tech/what-is-an-artificial-neural-network/
9. Li, Yang, et al. "Hashtag recommendation with topical attention-based LSTM." Coling, 2016.
10. Cho, Kyunghyun, et al. "Learning phrase representations using RNN encoder-decoder for statistical machine translation." *arXiv preprint arXiv: 1406.1078* (2014)
11. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. In *Advances in neural information processing systems* (pp. 5998-6008).
12. Sundermeyer, M., Schlüter, R., & Ney, H. (2012). LSTM neural networks for language modeling. In *Thirteenth annual conference of the international speech communication association*.
13. Rong, Xin. "word2vec parameter learning explained." *arXiv preprint arXiv: 1411.2738* (2014).
14. Goldberg, Yoav, and Omer Levy. "word2vec Explained: deriving Mikolov et al.'s negative-sampling word-embedding method." *arXiv preprint arXiv: 1402.3722* (2014).
15. Levy, O., & Goldberg, Y. (2014, June). Dependency-based word embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)* (pp. 302-308).

16. Goldberg, Y. (2017). Neural network methods for natural language processing. *Synthesis Lectures on Human Language Technologies*, *10*(1), 1-309.
17. Kiperwasser, Eliyahu, and Yoav Goldberg. "Simple and accurate dependency parsing using bidirectional LSTM feature representations." *Transactions of the Association for Computational Linguistics* 4 (2016): 313-327.
18. Adler, Meni, and Michael Elhadad. "An unsupervised morpheme-based hmm for hebrew morphological disambiguation." *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2006.
19. Conneau, Alexis, et al. "Supervised learning of universal sentence representations from natural language inference data." *arXiv preprint arXiv: 1705.02364* (2017).
20. Zitouni, Imed, ed. *Natural language processing of semitic languages*. Berlin: Springer, 2014.

9 נספחים

9.1 הרצת מחברת Jupyter

Running Jupyter Notebooks With The Anaconda Python Distribution

One of the requirements here is Python, either Python 3.3 or greater or Python 2.7. The general recommendation is that you use the Anaconda distribution to install both Python and the notebook application.

The advantage of Anaconda is that you have access to over 720 packages that can easily be installed with Anaconda's conda, a package, dependency, and environment manager. You can follow the instructions for the installation of Anaconda here for Mac or Windows.

Is something not clear? You can always read up on the Jupyter installation instructions here.

Running Jupyter Notebook The Pythonic Way: Pip

If you don't want to install Anaconda, you just have to make sure that you have the latest version of pip. If you have installed Python, you will typically already have it.

What you do need to do is upgrading pip:

```
# On Windows

python -m pip install -U pip setuptools

# On OS X or Linux

pip install -U pip setuptools
```

Once you have pip, you can just run

```
# Python2

pip install jupyter

# Python 3

pip3 install jupyter
```

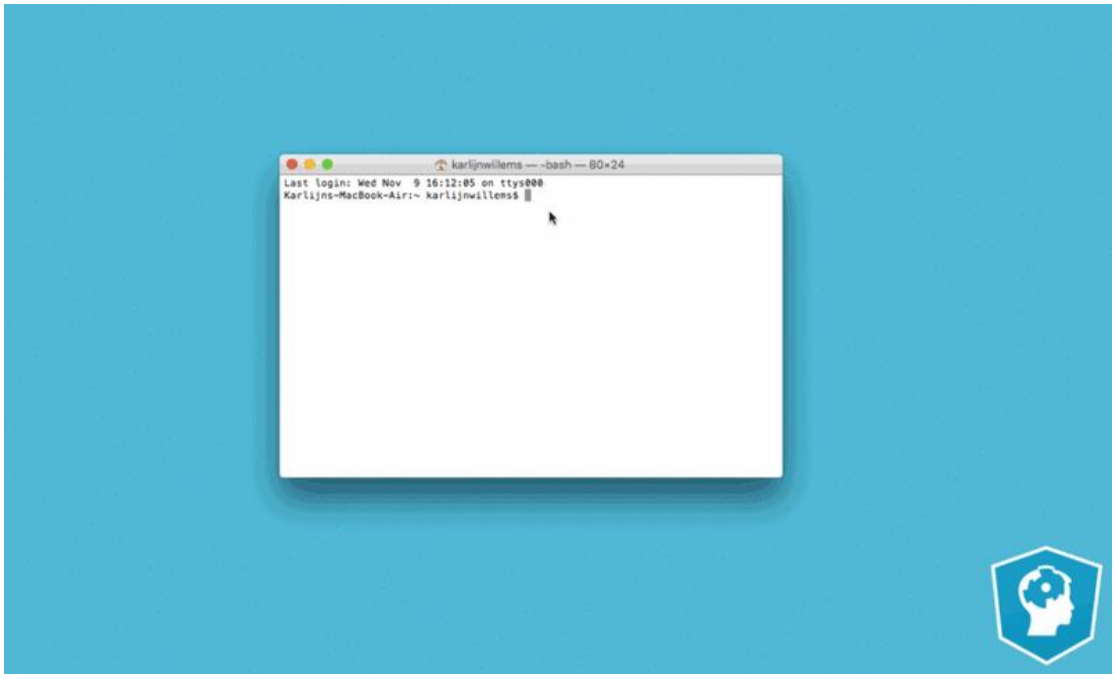
If you need more information about installing packages in Python, you can go to this page.

Getting Started With Jupyter Notebooks

Run the following command to open up the application:

```
jupyter notebook
```

Then you'll see the application opening in the web browser on the following address: http://localhost:8888. This all is demonstrated in the gif below:



The "Files" tab is where all your files are kept, the "Running" tab keeps track of all your processes and the third tab, "Clusters", is provided by IPython parallel, IPython's parallel computing framework. It allows you to control many individual engines, which are an extended version of the IPython kernel.

You probably want to start by making a new notebook.

You can easily do this by clicking on the "New button" in the "Files tab". You see that you have the option to make a regular text file, a folder, and a terminal. Lastly, you will also see the option to make a Python 3 notebook.

Note that this last option will depend on the version of Python that you have installed. Also, if the application shows python [conda root] and python [default] as kernel names instead of Python 3, you can try executing `conda remove _nb_ext_conf` or read up on the following Github issue and make the necessary adjustments.

Let's start first with the regular text file. When it opens up, you see that this looks like any other text editor. You can toggle the line numbers or/and the header, you can indicate the programming language you're writing, and you can do a find and replace. Furthermore, you can save, rename or download the file or make a new file.

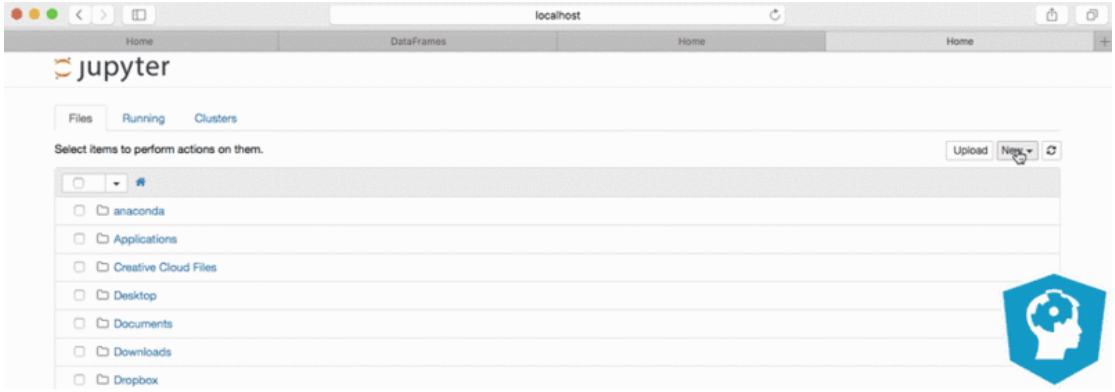
You can also make folders to keep all your documents organized together. Just press the "Folder" option that appears when you press the "New" button in your initial menu, and a new folder will appear in your overview. You can best rename the folder instantly, as it will appear as a folder with the name 'Untitled Folder'.

Thirdly, the terminal is there to support browser-based interactive terminal sessions. It primarily works just like your terminal or cmd application! Give in `python` into the terminal, press ENTER, and you're good to go.

Tip: if you would ever need a pure IPython terminal, you can type 'ipython' in your Terminal or Cmd. This can come in handy when, for example, you want to get more clear error messages than the ones that appear in the terminal when you're running the notebook application.

If you want to start on your notebook, go back to the main menu and click the "Python 3" option in the "Notebook" category.

You will immediately see the notebook name, a menu bar, a toolbar and an empty code cell:



You can immediately start with importing the necessary libraries for your code. This is one of the best practices that we will discuss in more detail later on.

After, you can add, remove or edit the cells according to your needs. And don't forget to insert explanatory text or titles and subtitles to clarify your code! That's what makes a notebook a notebook in the end.

9.2 מחברת הקוד

מומלץ להסתכל במחברת המצורפת או בקישור:

https://github.com/shplishka/project-finish/