Deep Learning

NLP part 2

RNN\LSTM\GRU

Bidirectional option

Return_sequence =True

Prediciting the next word

Transformers

חזרה- RNN

רשתות נוירונים חוזרות

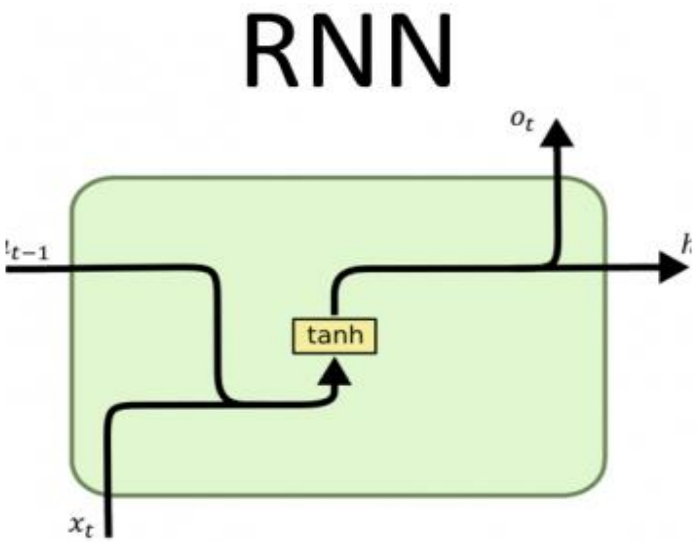
משמשות לעיבוד **רצפים (טקסט, דיבור, זמן וכו')**, עם **זיכרון פנימי** שמעבד מידע בשלבים.

כל צעד תלוי בקלט הנוכחי **ובמה שנלמד בשלבים הקודמים**.

סובלות מבעיית **הגרדיאנט הנעלם** (קשה לשמור מידע לאורך זמן רב).

מתאימות בעיקר לרצפים **קצרים יחסית** כמו משפטים קצרים.

פחות יעילות מטכניקות מודרניות כמו LSTM ו-Transformer.



LSTM – Long Short-Term Memory

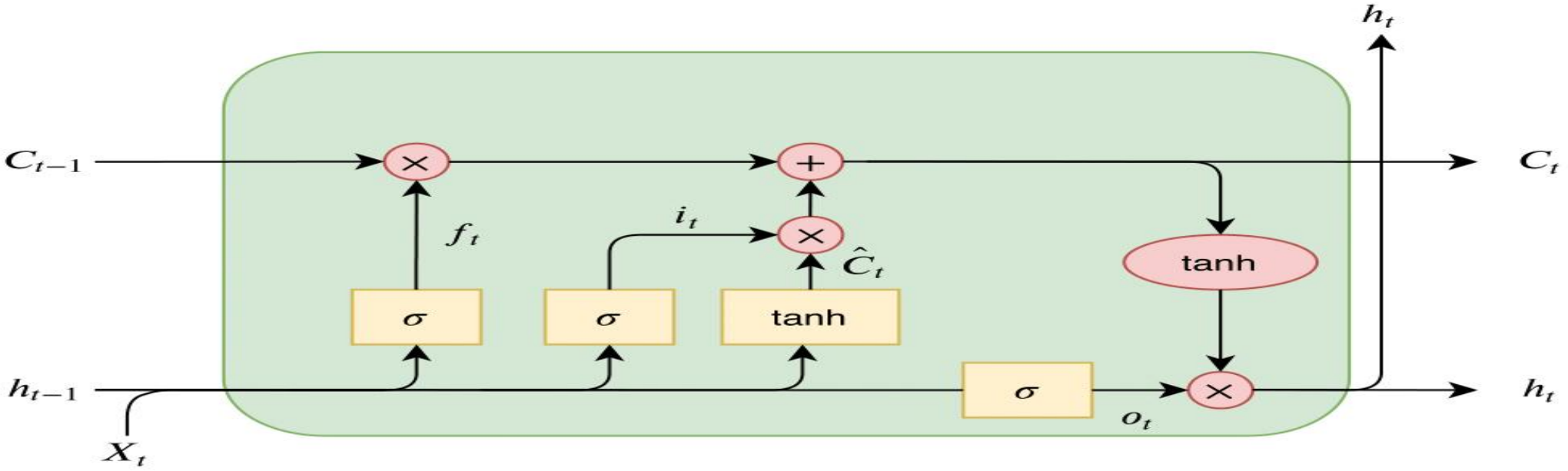
פותרות את בעיית **הגרדיאנט הנעלם** של RNN ע"י **שימוש בזיכרון לטווח ארוך**.

כוללות **שערים** ((Gatesלשליטה במה לזכור ומה לשכוח בכל שלב.

יכולות **לזכור מידע לאורך רצפים ארוכים**, מה שהופך אותן למתאימות יותר לטקסטים ארוכים.

משמשות הרבה ב- NLP, תרגום מכונה, זיהוי דיבור ומערכות המלצה.

כבדות יותר חישובית מ- RNN, ולכן פחות יעילות ממודלים כמו Transformer



GRU

- **פותר את בעיית הגרדיאנט הנעלם של RNN,** בדומה ל LSTM, אך בצורה פשוטה יותר.
- **משתמש בשני שערים בלבד** (Reset Gate & Update Gate) לשליטה על זרימת המידע.
- **לרוב מהיר יותר מ LSTM,** כי יש פחות פרמטרים לחישוב.
- **שומר מידע לטווחים ארוכים** ומתאים למשימות כמו עיבוד טקסט, זיהוי דיבור ותרגום מכונה.

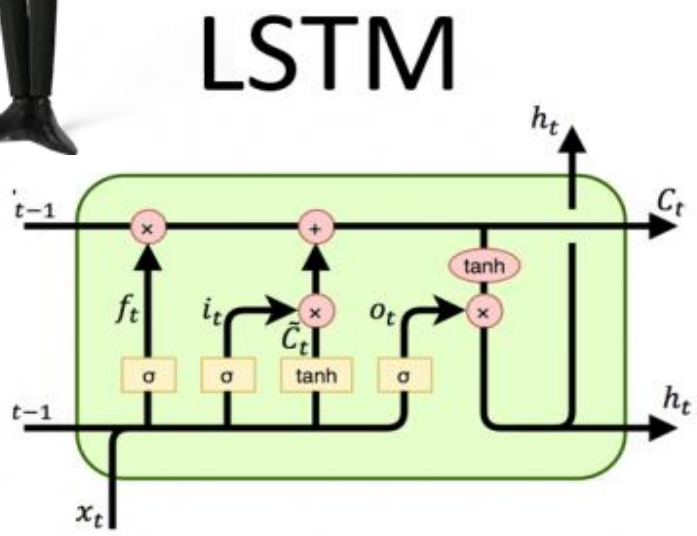
נוטה להיות מדויק פחות מ LSTM במקרים בהם נדרש זיכרון מאוד ארוך (טקסטים ארוכים),

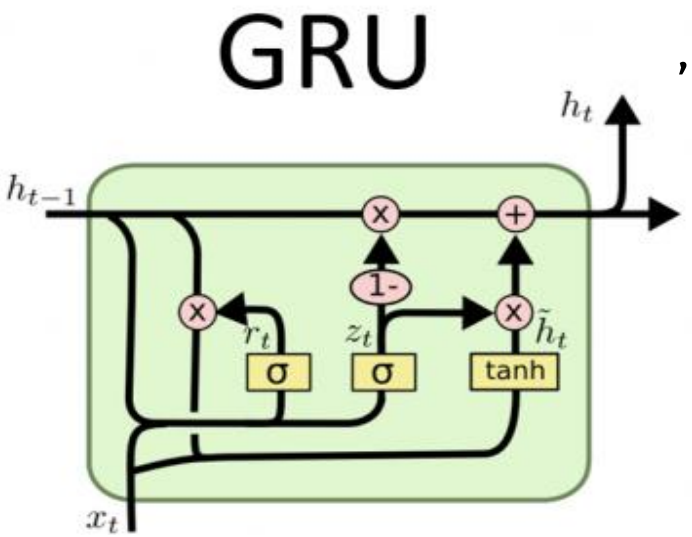
אך לרוב מספק ביצועים דומים בפחות חישובים.

- **משמש כחלופה קלה יותר ל LSTM**

ומשתלב היטב ב NLP, חיזוי סדרות זמן,

ויישומים מבוססי רצף .





TECHNION
Azrieli Continuing Education and External Studies Division

Return_sequences =True

מחזיר את כל השלבים ברצף

- כאשר מוגדר **return_sequences=False** (ברירת המחדל), השכבה מחזירה **רק את השלב האחרון** של ה RNN, -כלומר **וקטור יחיד לכל הרצף**.
- כאשר **return_sequences=True**, השכבה מחזירה **את כל השלבים** לאורך הרצף, כלומר **מטריצה** שבה כל צעד בזמן (timesteps) נשמר.
- **שימושי כאשר מעבירים את הפלט לשכבת RNN נוספת**, למשל ב **Stacked RNN / LSTM / GRU** -או במודלים כמו **Attention**.

```
model = Sequential([
    Embedding(input_dim=len(word_index)+1,

              output_dim=12),
    tf.keras.layers.SimpleRNN(16, return_sequences=True),
    tf.keras.layers.SimpleRNN(16, ),
```

TECHNION
Azrieli Continuing Education and External Studies Division

Bidirectional

– Bidirectional RNN / LSTM / GRU עיבוד רצף בשני הכיוונים

- **מעבד מידע קדימה ואחורה בו-זמנית**, כדי להבין את ההקשר מכל הכיוונים.
- **מתאים במיוחד למשימות NLP**, שבהן מילה יכולה לקבל משמעות שונה בהתאם להקשר שלה במשפט.
- **נפוץ ביישומים כמו תרגום מכונה, ניתוח סנטימנט, Named Entity Recognition וזיהוי דיבור**.
- **משפר דיוק** בהשוואה ל- RNN חד-כיווני, כי המודל רואה גם את העתיד של הרצף.
- **כבד יותר חישובית** כי הוא כולל **שני RNNs נפרדים** (אחד קדימה ואחד אחורה), אך לרוב שווה את זה.

שיטת עבודה:

במקום לעבד את המשפט **משמאל לימין בלבד**, כמו RNN רגיל, מודל דו-כיווני **מעבד גם מימין לשמאל** בו-זמנית.

כך הוא יכול להבין טוב יותר **תלות רחוקה** במשפטים מורכבים.

```
model = Sequential([
    Embedding(input_dim=len(word_index)+1, output_dim=12),
    # RNN דו-כיווני עם 16 יחידות, מחזיר רצפים להמשך עיבוד
    Bidirectional(SimpleRNN(16, return_sequences=True)),
     # RNN דו-כיווני נוסף, הפעם מחזיר רק את התוצאה האחרונה
    Bidirectional(SimpleRNN(16)),
```

Transformers

- **מודל לעיבוד רצפים (טקסט, דיבור, תמונה) ללא שימוש ב-RNN**או.CNN
- מבוסס על ,**Self-Attention** שמאפשר למודל להתמקד במילים החשובות ביותר בכל שלב.
- **יעיל בהרבה מ** → **RNN/LSTM/GRU**-ניתן לבצע חישובים במקביל, מה שמאפשר עיבוד מהיר של רצפים ארוכים.
- נמצא בבסיס של מודלים מתקדמים כמו **BERT, GPT, T5** ו - **Whisper**.
- משמש ביישומים כמו **תרגום מכונה, ניתוח טקסט, צ'אטבוטים ויצירת תוכן אוטומטית**

TECHNION
Azrieli Continuing Education and External Studies Division

Transformers

בנוי מ 3 שכבות:

1. **Self-Attention מנגנון תשומת הלב העצמית**

- מאפשר למודל **להתמקד במילים החשובות ביותר** בהקשר של כל מילה במשפט.
- כל מילה משווה את עצמה **לכל שאר המילים** במשפט כדי להבין את ההקשרים ביניהן.
- משמש בליבת הטרנספורמר – בפרט בתוך **Multi-Head Attention**.

2. **Feed Forward Network רשת (Fully Connected)**

- שכבה שמופעלת על כל טוקן (מילה) בנפרד, אחרי שלב ה-Self-Attention.
- מבצעת **למידה עמוקה יותר** על הייצוג של כל מילה לאחר שהתחשבנו בהקשר שלה.
- לרוב מורכבת משתי שכבות Dense עם פונקציית אקטיבציה **ReLU**

3. **Positional Encoding קידוד מיקום המילים**

- בניגוד ל- RNN, שבו סדר המילים מובנה, הטרנספורמר **לא רואה סדר מטבעו**.
- כדי להתמודד עם זה, מוסיפים **וקטור ייחודי לכל מילה שמייצג את המיקום שלה במשפט**.
- מבטיח שהמודל יידע להבין **איזה מילה הופיעה קודם ואיזו אחריה**.

Transformers

הטרנספורמר מורכב מ- **Stackשל בלוקים באלה** - כל בלוק מכיל

Self-Attention + Feed Forward + Positional Encoding.

ב- Encoder המודל מקודד את המשפט כולו, וב-**Decoder** הוא מחלץ את הפלט (למשל, בתרגום מכונה).