מבוא ללמידה עמוקה- תרגיל 2
דניאל בראון 311340723, אליהו סטרוגו ת.ז. 305589269

.1

a. נגדיר:

$$f_1(x,y,z) = x + y$$
$$f_2(x,y,z) = 2x$$
$$f_3(x,y,z) = z$$

נשים לב כי

$$f(x+y, 2x, z) = f\big(f_1(x,y,z), f_2(x,y,z), f_3(x,y,z)\big)$$

וכמו כן כי $f: \mathbb{R}^3 \to \mathbb{R}^d$ עבור $d \in \mathbb{N}$ כלשהו.
נשתמש בכלל השרשרת:

$$f\big(g(x)\big)' = f'\big(g(x)\big) \cdot g'(x)$$

וניישם אותו על המקרה שלנו

$$= \left[\frac{\partial}{\partial f_1} f(f_1,f_2,f_3), \frac{\partial}{\partial f_2} f(f_1,f_2,f_3), \frac{\partial}{\partial f_3} f(f_1,f_2,f_3)\right] \cdot \begin{bmatrix} \frac{\partial f_1}{\partial x} \\ \frac{\partial f_2}{\partial x} \\ \frac{\partial f_3}{\partial x} \end{bmatrix}$$

נשים לב כי

$$\frac{\partial f_1}{\partial x} = 1$$
$$\frac{\partial f_2}{\partial x} = 2$$
$$\frac{\partial f_3}{\partial x} = 0$$

ולכן:

$$\frac{\partial}{\partial x} f = \left[\frac{\partial}{\partial f_1} f(f_1,f_2,f_3), \frac{\partial}{\partial f_2} f(f_1,f_2,f_3), \frac{\partial}{\partial f_3} f(f_1,f_2,f_3)\right] \cdot \begin{bmatrix} 1 \\ 2 \\ 0 \end{bmatrix}$$
$$= \frac{\partial}{\partial f_1} f(f_1,f_2,f_3) + 2\frac{\partial}{\partial f_2} f(f_1,f_2,f_3)$$

כנדרש.

b. נכתוב את כלל השרשרת לשרשור של פונקציות:

$$\Big(f_1\Big(f_2\big(\dots f_n(x)\big)\Big)\Big)' = \prod_{i=1}^{n} \frac{\partial}{\partial f_{i+1}\Big(f_{i+2}\big(\dots f_n(x)\big)\Big)} f_i$$
$$= \frac{\partial}{\partial f_2\Big(f_3\big(\dots f_n(x)\big)\Big)} f_1 \cdot \frac{\partial}{\partial f_3\Big(f_4\big(\dots f_n(x)\big)\Big)} f_2 \cdot \dots \cdot \frac{\partial}{\partial x} f_n$$

שזוהי גזירה זהה לזו שביצענו באלגוריתם ה-$Back\ Propogation$.

c. הייתי משתמש ב-$ReLu$ כפונקציית אקטיבציה בכל שכבה (בתוך כל $f_i$) מכיוון שהנגזרות שלה יציבות נומרית. נפרט: אם נתקל בבעיית $vanishing\ gradient$ נרצה להשתמש ב-$ReLu$ מכיוון שהנגזרת שלו היא או 0 או 1. נקבל נגזרת 1 לכל ערך $x > 0$, וערך 0 לכל $x < 0$. לעומתו, למשל, פונקציית האקטיבציה $sigmoid$ נהיית רוויה בקצוות, ולכן הנגזרת שלה מקבלת 1 ב-0, וערך קטן מ-1 בכל נק' אחרת. אם נחליט להשתמש בסיגמואיד, אזי כמעט כל ערך שנקבל (למעט 0) יהיה קטן מ-1, ולכן כל ערך כזה יגביר לבעיית ה=$vanishing\ gradient$. באופן כזה $ReLu$ תורם יותר לייצוב נומרי של הרשת.

d. נגזור את הביטוי $f = f_1\left(x, f_2\left(x, f_3\left(\dots f_{n-1}\left(x, f_n(x)\right)\right)\right)\right)$:

$$f' = \left(\frac{\partial f_1}{\partial x}, \frac{\partial f_1}{\partial f_2}\cdot\frac{\partial f_2}{\partial x}\right) = \left(\frac{\partial f_1}{\partial x}, \frac{\partial f_1}{\partial f_2}\cdot\left(\frac{\partial f_2}{\partial x}, \frac{\partial f_2}{\partial f_3}\cdot\frac{\partial f_3}{\partial x}\right)\right)$$
$$= \left(\frac{\partial f_1}{\partial x}, \frac{\partial f_1}{\partial f_2}\cdot\left(\frac{\partial f_2}{\partial x}, \frac{\partial f_2}{\partial f_3}\cdot\left(\frac{\partial f_3}{\partial x}, \frac{\partial f_3}{\partial f_4}\right)\dots\frac{\partial f_n}{\partial x}\right)\right)$$

e. נסביר מה היתרון של ביטוי זה על פני 'סתם' הרכבה של $f, g, h$. נעשה זאת ע"י גזירת הביטוי:

$$\frac{\partial f\left(x+g\left(x+h(x)\right)\right)}{\partial x} = \frac{\partial f\left(x+g\left(x+h(x)\right)\right)}{\partial\left(x+g\left(x+h(x)\right)\right)}\cdot\frac{\partial\left(x+g\left(x+h(x)\right)\right)}{\partial x}$$

נגזור את $\frac{\partial\left(x+g\left(x+h(x)\right)\right)}{\partial x}$:

$$\frac{\partial\left(x+g\left(x+h(x)\right)\right)}{\partial x} = \frac{\partial x}{\partial x} + \frac{\partial\left(g\left(x+h(x)\right)\right)}{\partial x} = 1 + \frac{\partial\left(g\left(x+h(x)\right)\right)}{\partial x}$$

נגזור את $\frac{\partial\left(g\left(x+h(x)\right)\right)}{\partial x}$:

$$\frac{\partial\left(g\left(x+h(x)\right)\right)}{\partial x} = \frac{\partial\left(g\left(x+h(x)\right)\right)}{\partial\left(x+h(x)\right)}\cdot\frac{\partial\left(x+h(x)\right)}{\partial x} = \frac{\partial\left(g\left(x+h(x)\right)\right)}{\partial\left(x+h(x)\right)}\cdot\left(1+\frac{\partial h(x)}{\partial x}\right)$$

סה"כ נקבל:

$$\frac{\partial f\left(x+g\left(x+h(x)\right)\right)}{\partial x} = \frac{\partial f\left(x+g\left(x+h(x)\right)\right)}{\partial\left(x+g\left(x+h(x)\right)\right)}\cdot\left(1+\frac{\partial\left(g\left(x+h(x)\right)\right)}{\partial\left(x+h(x)\right)}\cdot\left(1+\frac{\partial h(x)}{\partial x}\right)\right)$$

כעת נתבונן בביטוי של הרכבה רגילה:

$$\frac{\partial f\left(g\left(h(x)\right)\right)}{\partial x} = \frac{\partial f\left(g\left(h(x)\right)\right)}{\partial g(h(x))}\cdot\frac{\partial g\left(h(x)\right)}{\partial x} = \frac{\partial f\left(g\left(h(x)\right)\right)}{\partial g\left(h(x)\right)}\cdot\left(\frac{\partial g\left(h(x)\right)}{\partial h(x)}\cdot\frac{\partial h(x)}{\partial x}\right)$$

נשים לב כי ביטוי זה יהיה קטן יותר מהביטוי הקודם במידה וכל הנגזרות קטנות מ-0, ולכן הוא פחות יציב נומרית. לכן הרכבת פונקציות מהצורה שנתונה בשאלה (שהיא בעצם $skip\ connection$) היא יציבה יותר מבחינה נומרית.

2.

a. עבור משימת זיהוי דיבור (*Speech Recognition*) נשתמש ב-$RNN$ (*recurrent neural network*) בשיטת *many-to-many*. משימה כזו מקבלת קלט בגודל משתנה וצריכה לפלוט פלט בגודל משתנה. בנוסף, $RNN$ מאפשר לייצר פרדיקציה כל הזמן תוך כדי הריצה על הקלט. ארכיטקטורה של $RNN$ בשיטה זו יאפשר להתמודד עם דרישות אלו.

b. עבור משימת מענה על שאלות (*Answer Questions*) נשתמש ב-$RNN$. נשים לב לצורך משימה זו נצטרך להשתמש בשיטת *many-to-many*, בדומה לזיהוי דיבור. באופן דומה לניתוח רגש, מכיוון שבמענה על שאלה חשוב להתחשב בהיסטוריה של הקלט שנתקבל, נשתמש ב-$RNN$.

c. עבור משימת ניתוח רגש (*Sentiment Analysis*) נשתמש ב-$RNN$, בדומה למימוש התרגיל. ראשית נשים לב שזוהי משימה מסוג *many-to-few*, כאשר הקלט (ביקורות, למשל) אינו מוגבל בגודל, והוא ממופה למס' סופי וקבוע של מחלקות-רגשות. מכיוון שמשימה זו מקבלת גודל לא קבוע של קלט, ומכיוון שהיא נזקקת לתת פרדיקציה תוך התחשבות מתמדת בהיסטוריה של הקלט שנתקבל, הארכיטקטורה המתאימה כאן היא $RNN$.

d. עבור משימת סיווג תמונות (*Image Clasiffication*) נשתמש ב-$CNN$ (רשת נוירונים קונבולוציונית). כאשר אנו ניגשים למשימת סיווג תמונות, נרצה 'להעניק' לרשת את היכולת להסתכל על התמונה בקנה מידה גדול יותר מאשר פיקסל (שהוא קנה המידה הבסיסי של המחשב). נוכל לתת לרשת את האפשרות לעשות זאת אם נבצע קונבולוציה על התמונה, שהיא פונקציה שמבוצעת על מס' פיקסלים. בנוסף, קונבולוציה היא פעולה שהיא אי-ווריאנטית למיקום/סדר בקלט (כלומר אין חשיבות למיקום של החתול בתמונה, אלא רק להיותו שם).

$e$. עבור משימת תרגום של מילה אחת (*Single Word Translation*) נשתמש ב-*Transformer network*. נזכר כי בדומה ל-*RNN*, מודל זה נועד כדי להתמודד עם *sequential data*, ולכן זה מתאים לעיבוד שפה טבעית.

3.

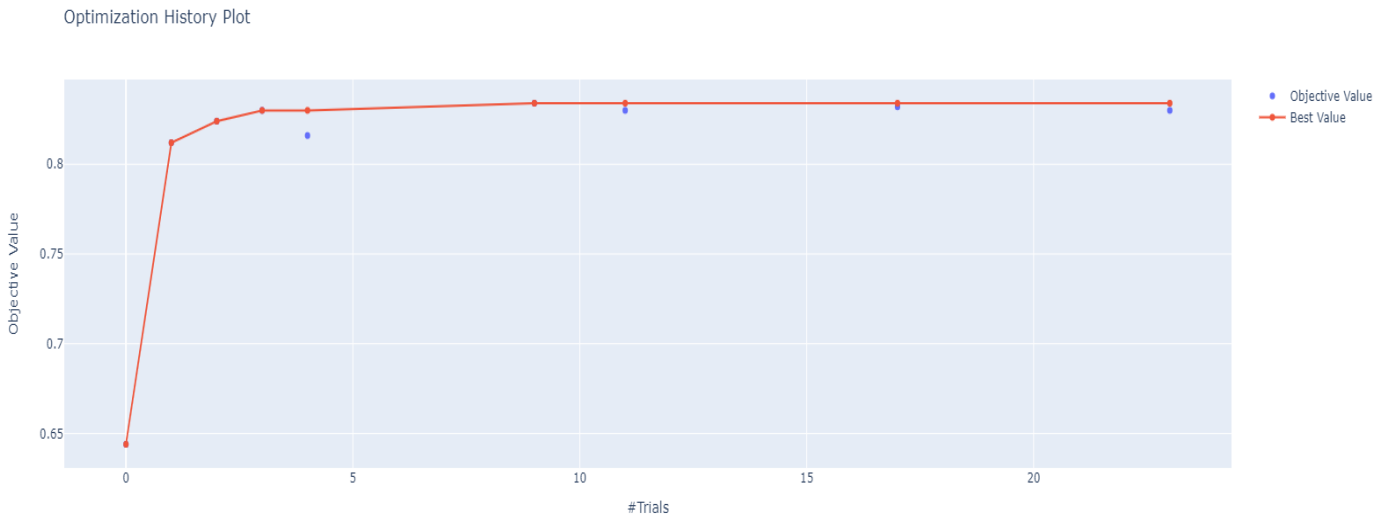
א. למשימה זו נוכל להשתמש ב-$Auto-Encoder$. נפרט באיזה אופן:
נייצר שני מרחבים לטנטים בעלי אותו המימד- מרחב לטנטי אחד של התמונות ומרחב לטנטי שני של המשפטים.
בהינתן משפט, נעשה לו *Encode* וכך נקבל וקטור שמייצג את המשפט במרחב הלטנטי.
אם נמצא העתקה מתאימה מהמרחב הלטנטי של המשפטים למרחב הלטנטי של התמונות, נקבל את הרצוי.
נוכל להפעיל $Decoder$ שפולט תמונות בהינתן הוקטור המייצג את המשפט, וכך נקבל את התמונה הרצויה.

ב. הארכיטקטורה המתוארת, בצירוף שכבת *Attention*, תאפשר לנו לתמוך בתיאורי איזור של הקלט. באמצעות ארבעת הוקטורים המייצגים רביעים בכל תמונה, אנו יכולים לייצר יצוג וקטורי של התמונה שמתנהג באופן דומה לשכבת ה-*Attention*. נתאר את ה-$Q,V,K$ בשכבת ה-*Attention* שנצרף: $V$ יהיה אוצר המילים (כוקטורים), $Q$ יהיה כל אחד מהרבעים הנתונים (כוקטורי יחידה) ו-$K$ יהיה ההסתברות לקבלת רביע מסוים בהינתן מילה, כלומר $p(q|v)$, וניקח את הרביע שממקסם את ההסתברות.

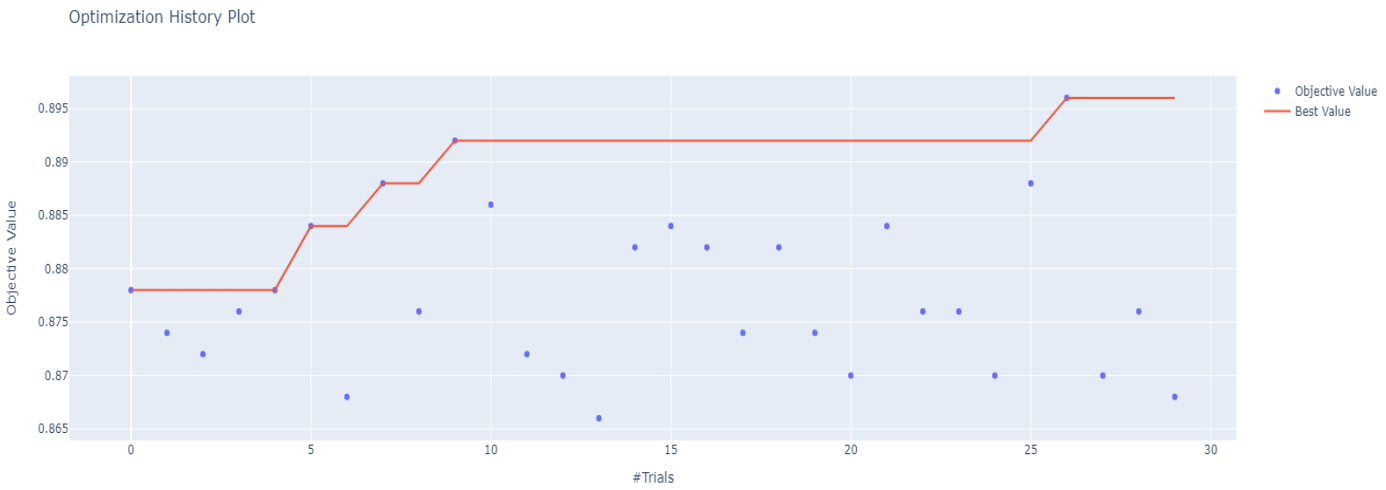
## חלק מעשי

1. תחילה נסינו לבדוק את הביצועים של הרשתות בקצוות – 64 ו 128 *hidden-states* . כאשר ראינו שיש *overfit* עבור 128 מצבים נסינו למצוא מספר מצבים בטווח 64-128 שעבורו נקבל את ה *accuracy* הטוב ביותר על ה *validation set* . כדי לעשות זאת השתמשנו באלגוריתם חיפוש Tree-Structured Parzen Estimator (TPE).
התוצאות שקיבלנו הן *RNN: 86 GRU: 107* ( כמובן שסביר שזה הוא לא פתור אופטימלי).

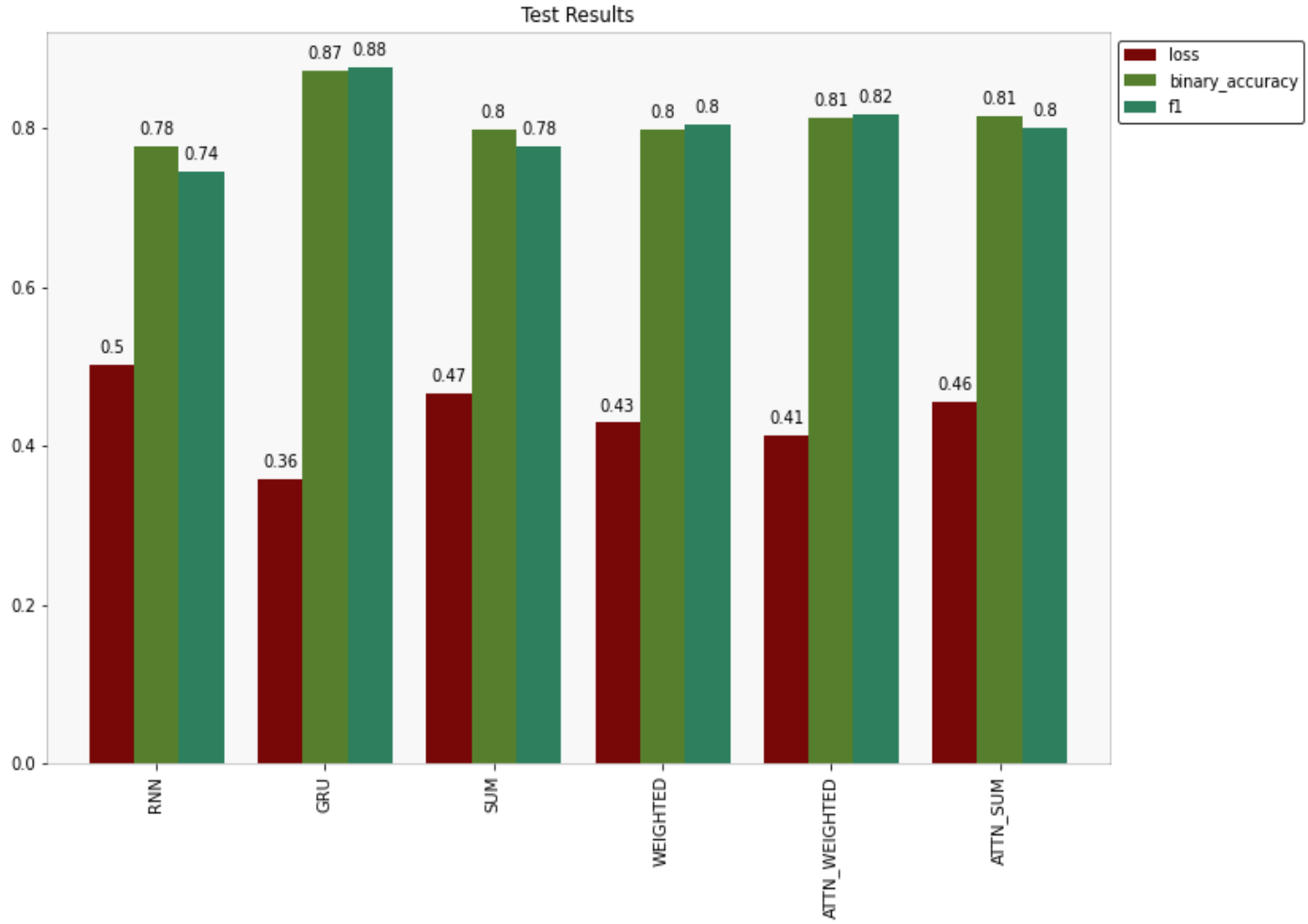
גרף החיפוש עבור *RNN* :



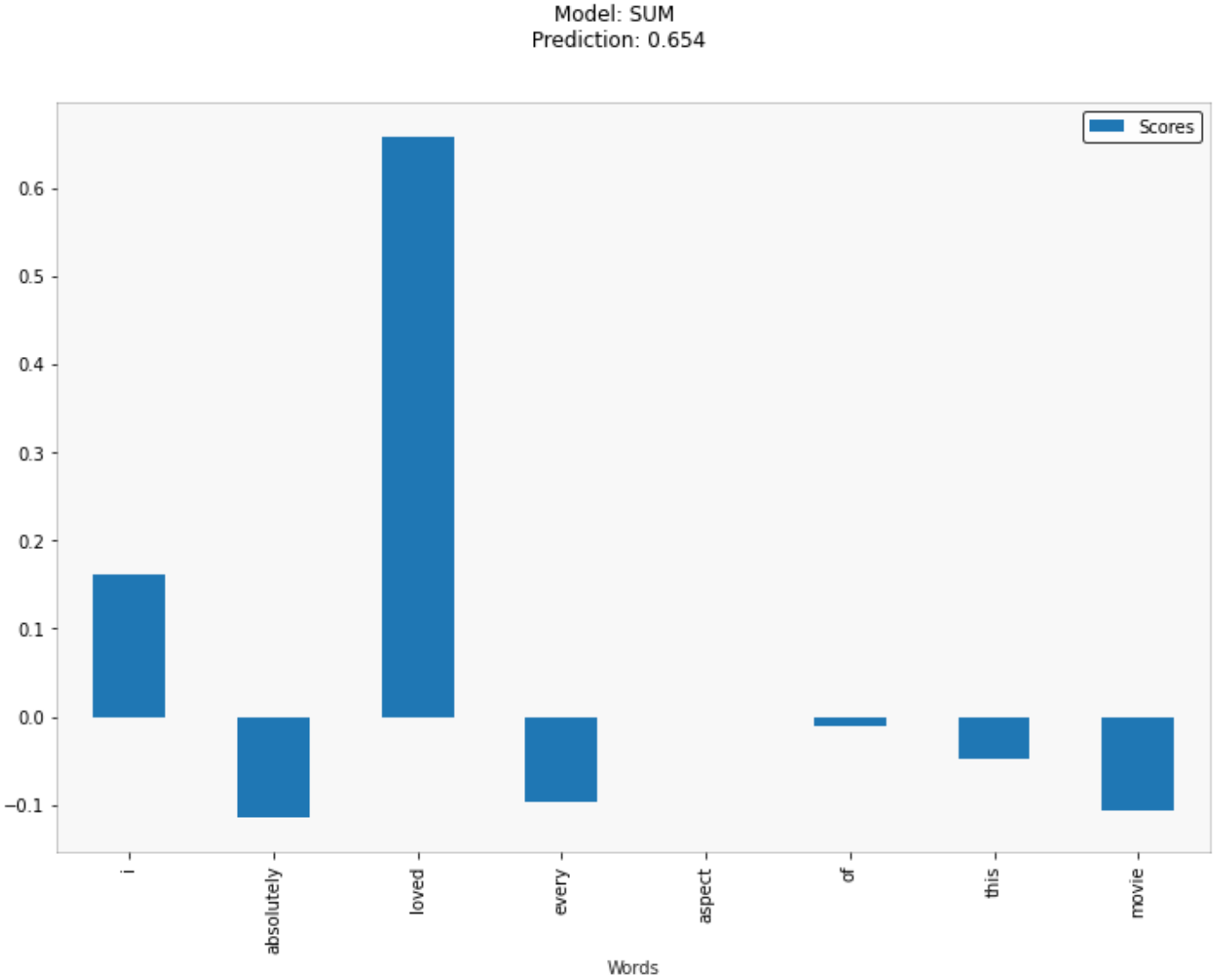

גרף החיפוש עבור *GRU* :



השוואה בין המודלים :

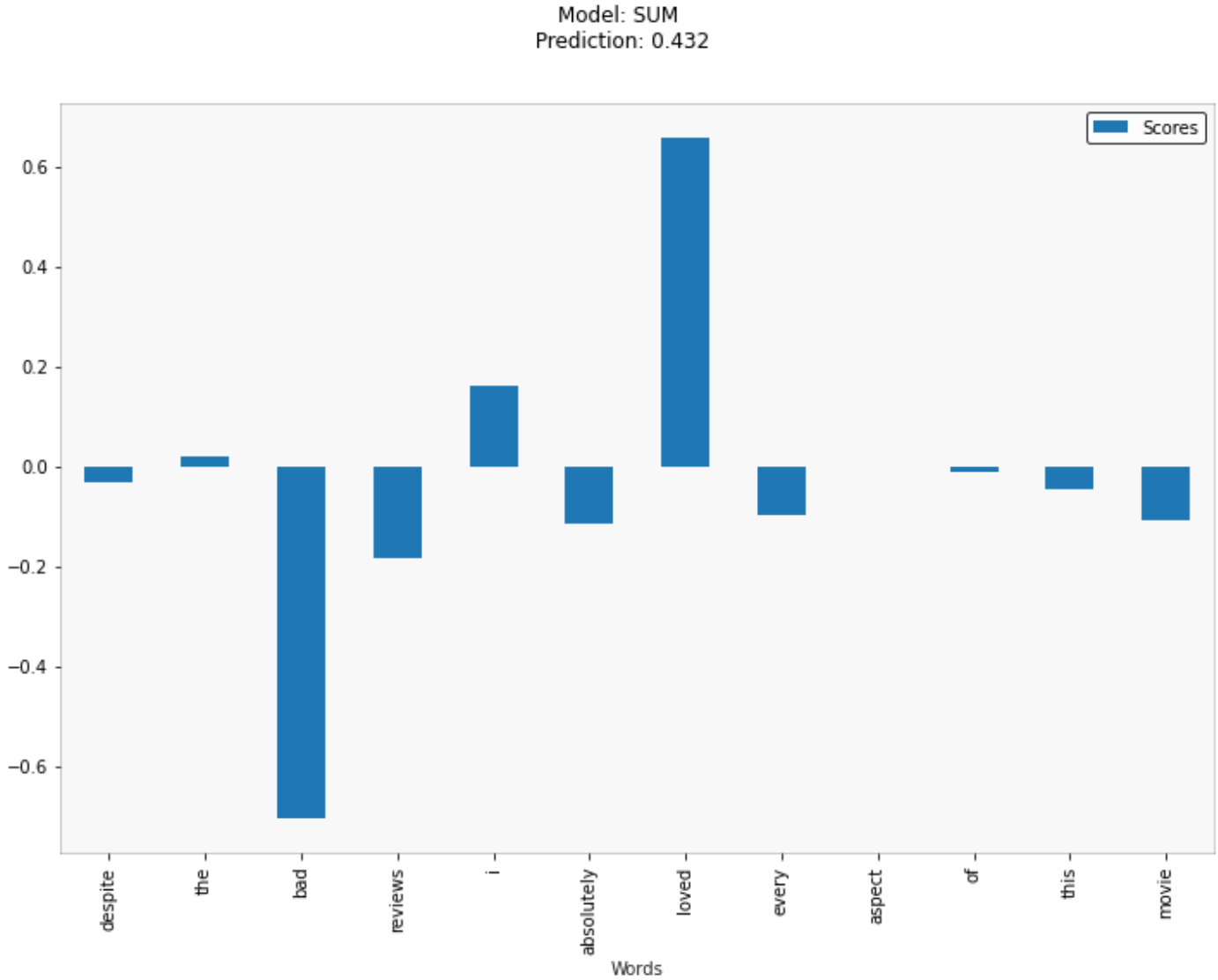


2. דוגמה לפרדיקציה נכונה עבור הביקורת : *" i absolutely loved every aspect of this movie"*



דוגמה לפרדיקציה לא נכונה עבור הביקורת:

*"despite the bad reviews i absolutely loved every aspect of this movie"*

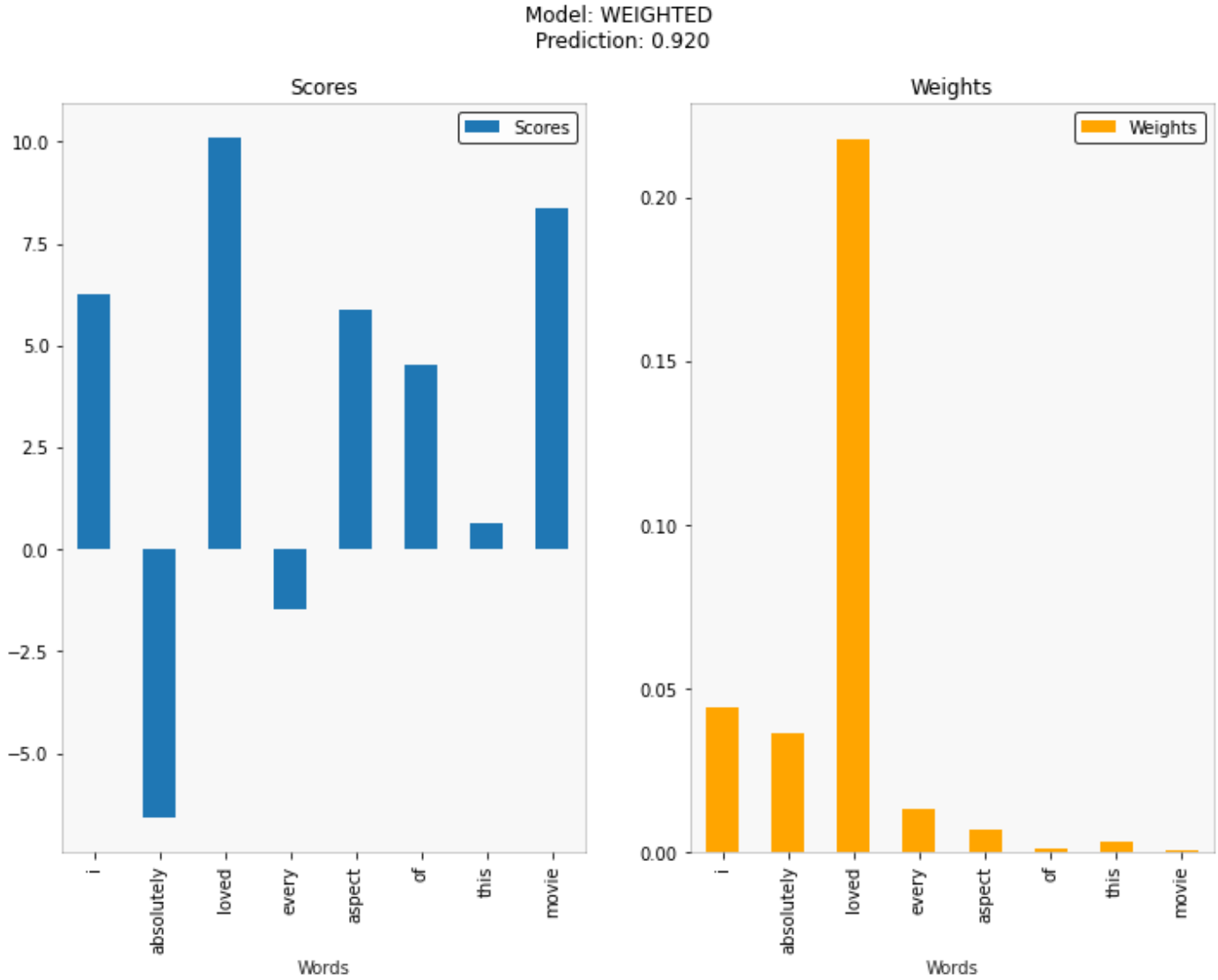



על ידי הוספה של משפט *"despite the bad reviews"* שאינו אמור להשפיע על העובדה שהביקורת חיובית הצלחנו לגרום למודל לשנות את הפרדיקציה מחיובית לשלילית.

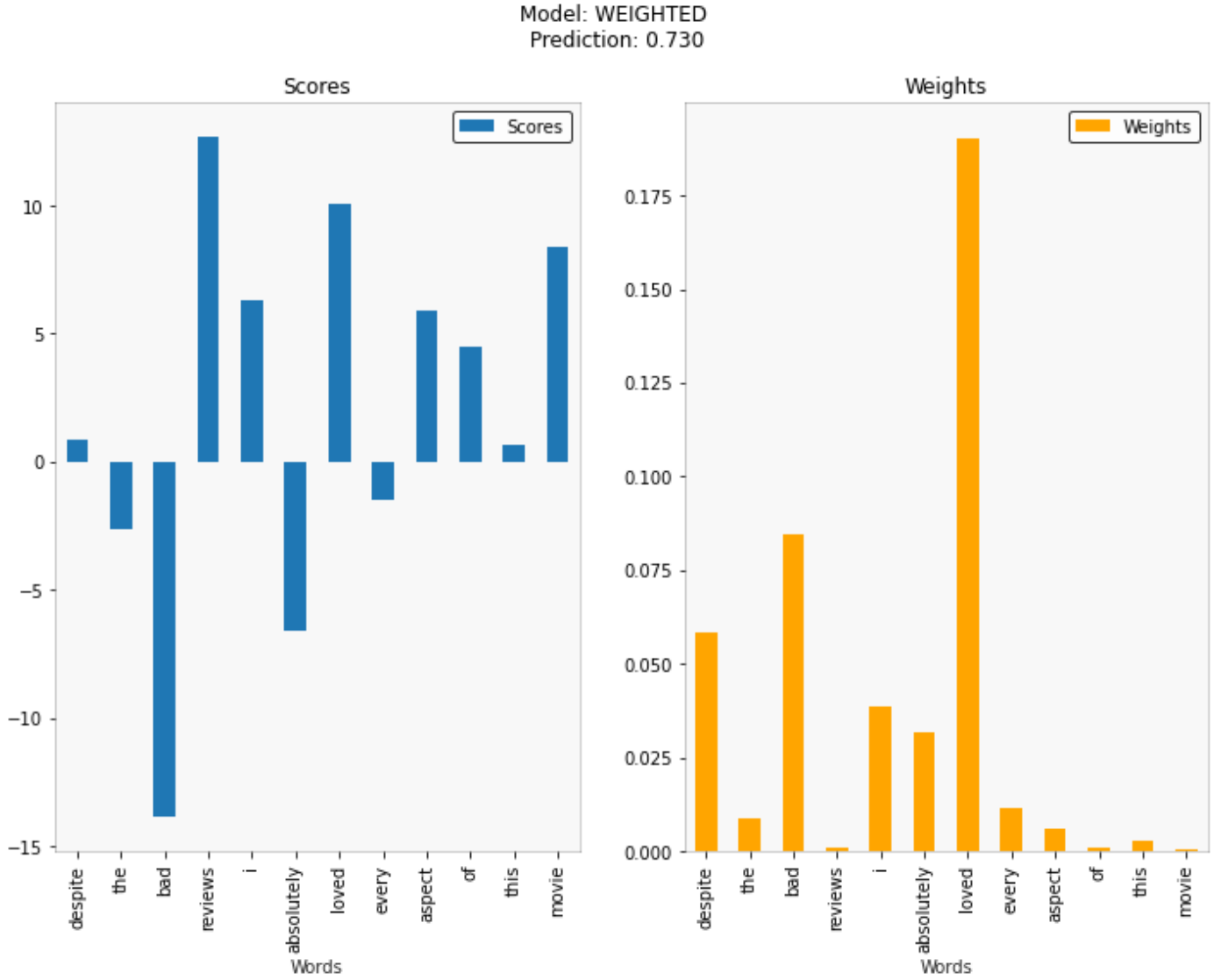
למודל אין יותר מדי יכולת להבין את ההקשר של המילה *bad* במפשט ולכן משום של-*bad* שיש ניקוד מאוד נמוך ביחס למילים האחרות, אז הסכום של הניקוד של כלל המילים יהיה קטן מספיק כדי לגרום לפרדיקציה שלילית.

.3

*"i absolutely loved every aspect of this movie*



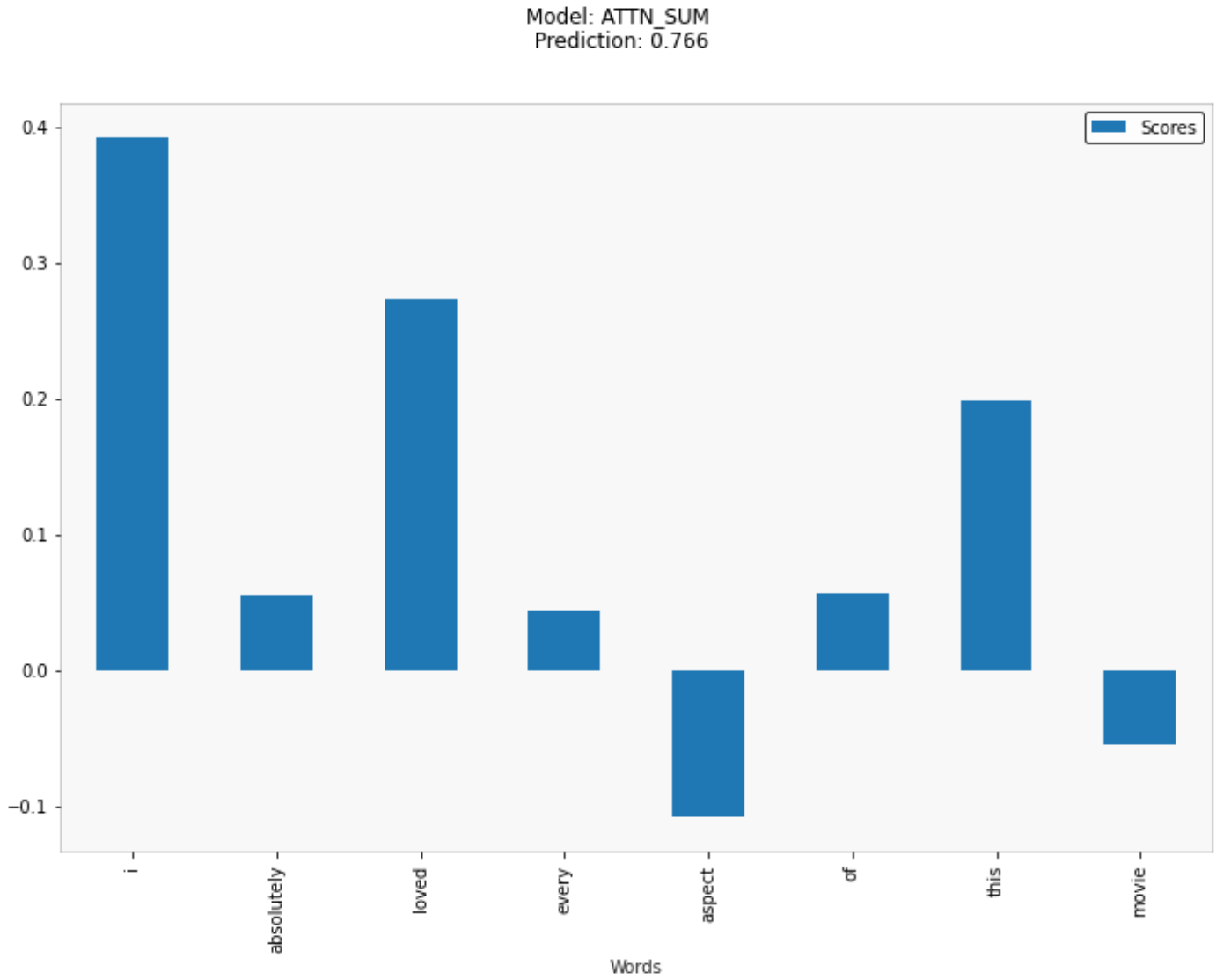
*"despite the bad reviews i absolutely loved every aspect of this movie"*

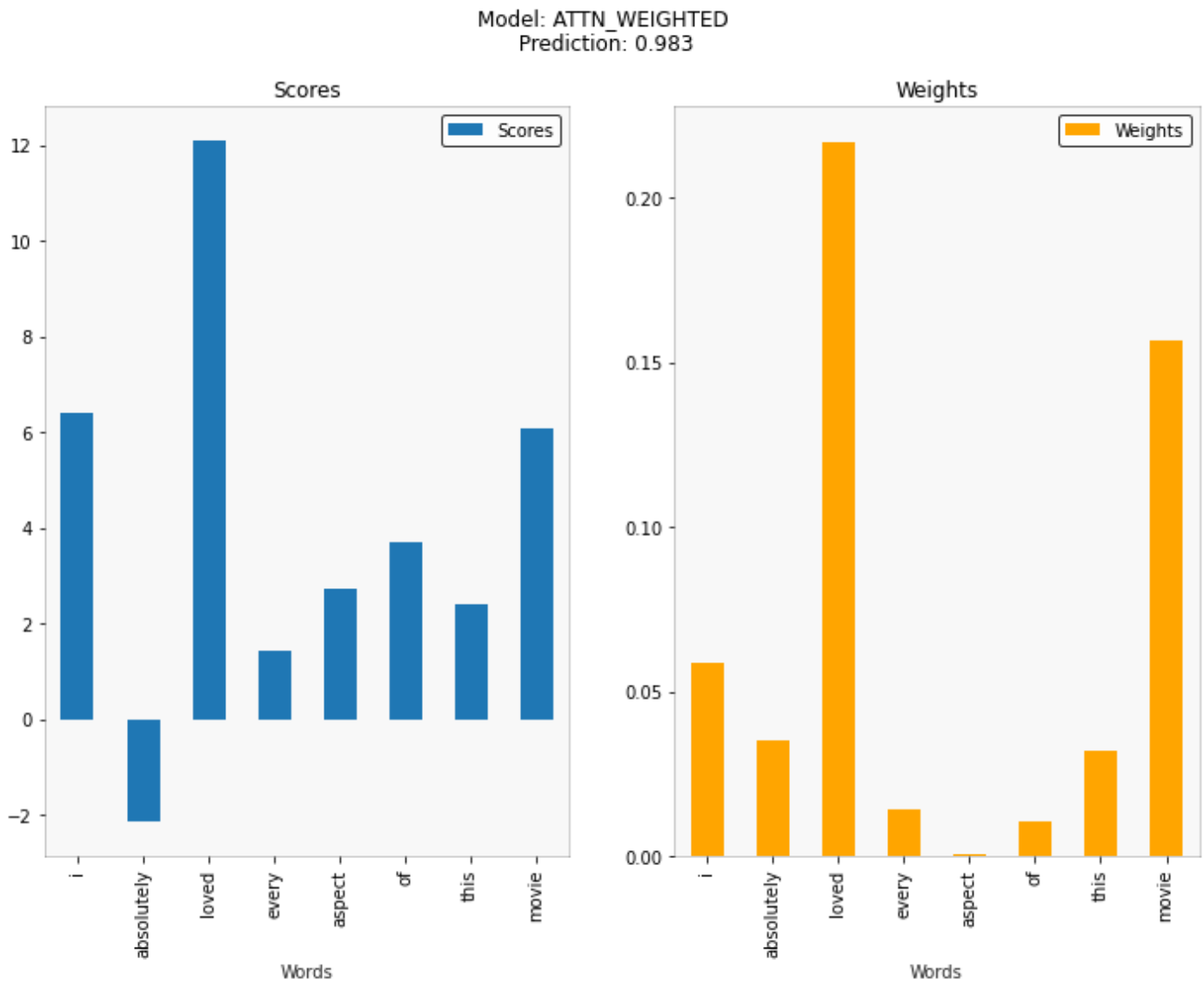



עבור הביקורת החיובית אנחנו אכן רואים רמת הביטחון של המודל בביקורת החיובית אכן גבוהה יותר, ובביקורת השנייה עי ידי משקול המודל מצליח להתגבר על הניקוד הנמוך של המילה *bad* בתחילת המשפט ולספק פרדיקציה נכונה.( למרות משקול גבוה יחסית למילה *bad* מה שיכול ללמד גם על המגבלות של המודל הזה ביחס להקשר של המילה הזו במשפט).

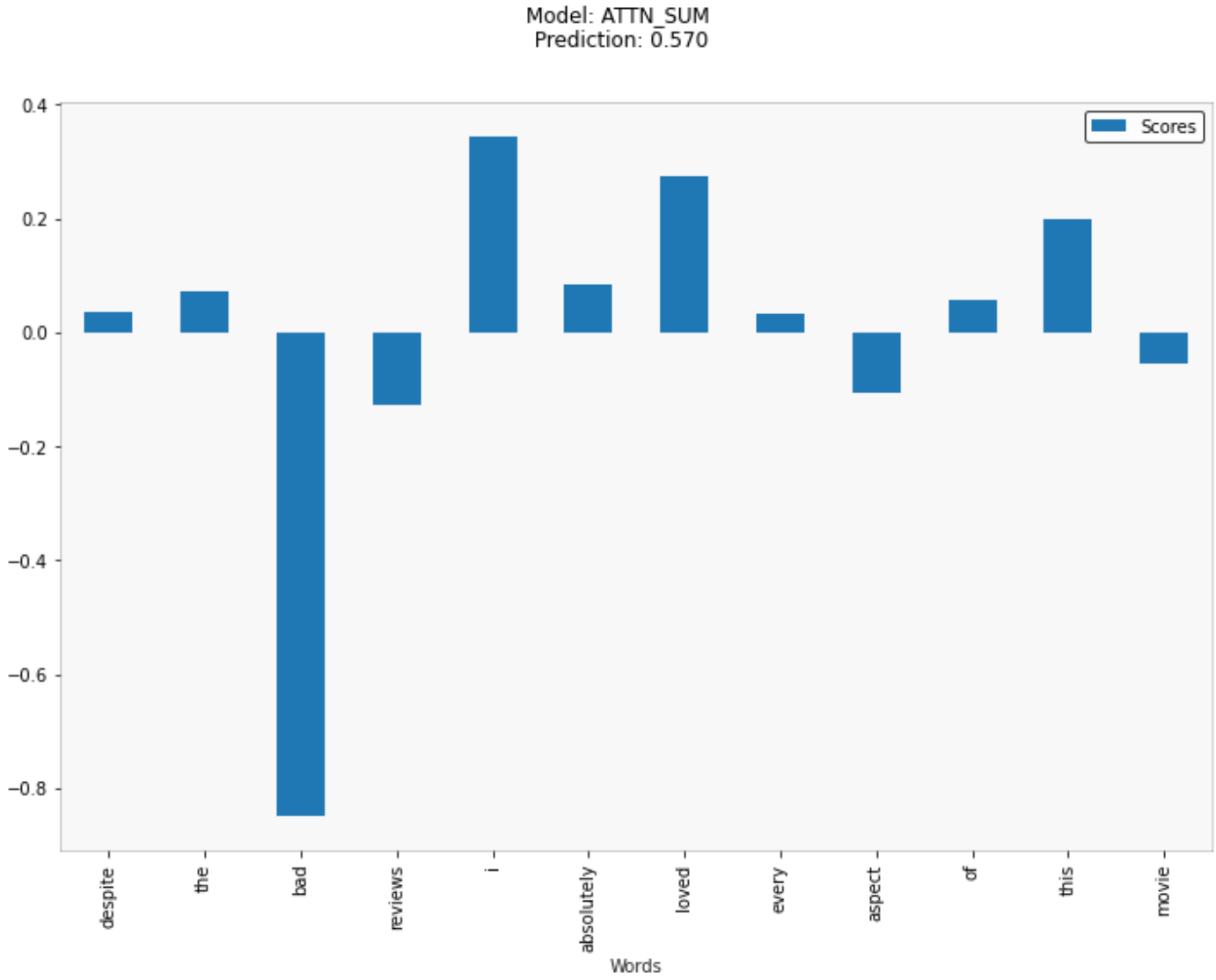
.5

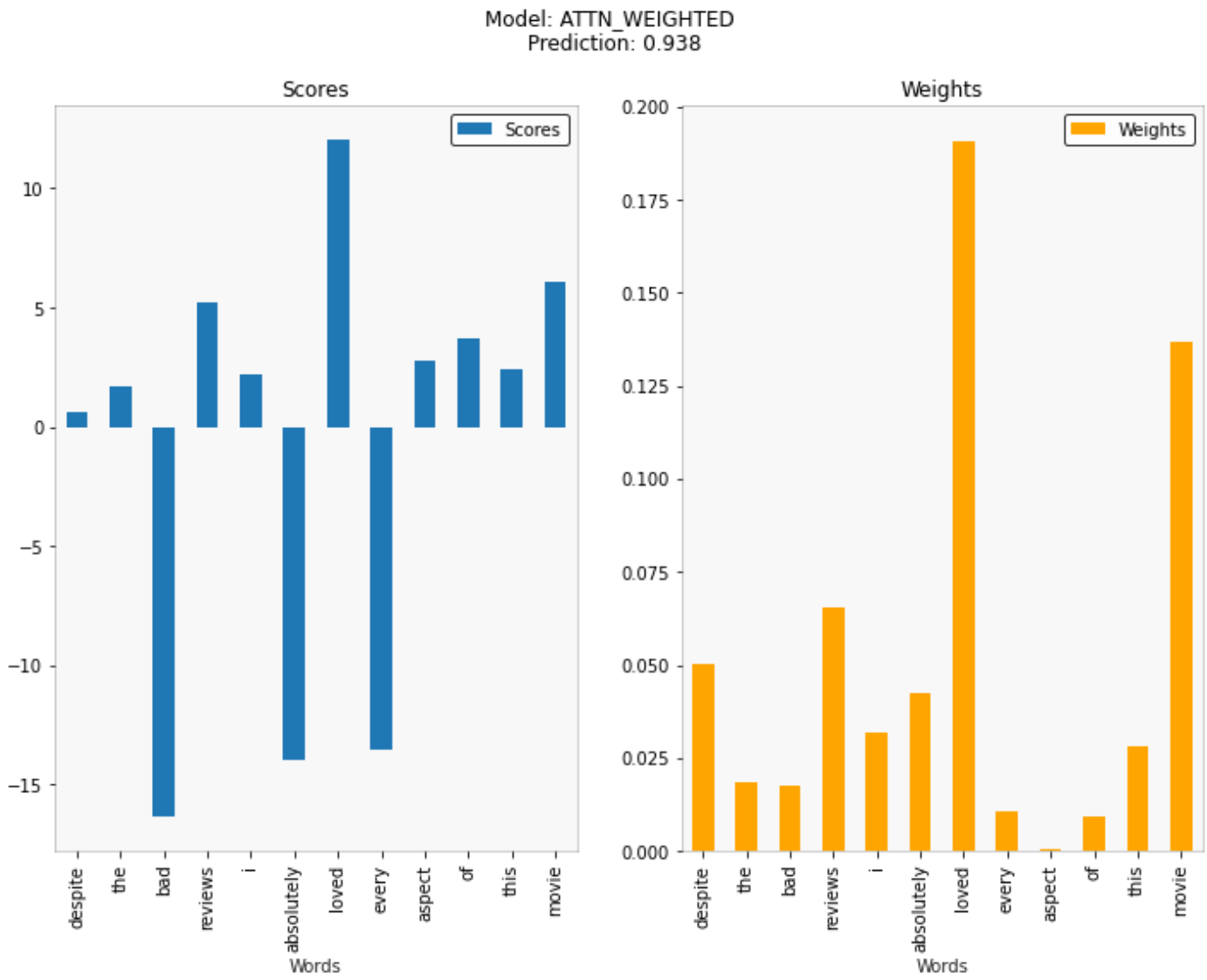
*"i absolutely loved every aspect of this movie*




Model: ATTN_WEIGHTED
Prediction: 0.983
Scores
Scores
12
10
8
6
4
2
0
−2
i
absolutely
loved
every
aspect
of
this
movie
Words
Weights
Weights
0.20
0.15
0.10
0.05
0.00
i
absolutely
loved
every
aspect
of
this
movie
Words

*"despite the bad reviews i absolutely loved every aspect of this movie"*




Model: ATTN_WEIGHTED
Prediction: 0.938
Scores
Scores
10
5
0
−5
−10
−15
despite
the
bad
reviews
i
absolutely
loved
every
aspect
of
this
movie
Words
Weights
Weights
0.200
0.175
0.150
0.125
0.100
0.075
0.050
0.025
0.000
despite
the
bad
reviews
i
absolutely
loved
every
aspect
of
this
movie
Words

כפי שניתן לראות, בשני המקרים ההוספה של *attention layer* משפרת את הביצועים.

העקרון העיקרי הוא היכולת של ה- *self-attention* הוא להצליב בין מילים במשפט שלנו ובכך לתת להן משמעות לפי ההקשר. השתמשו ב *attention weights* כדי לנסות לתאר את הקשר על פי המודל :