מבוא ללמידה עמוקה: תרגיל 2

8 בינואר 2021

מגישים:
מרינה רומנצ'וק, תז: 323255844
דיוויד פיץ, תז: 080173313

חלק 2- חלק תאורטי- שאלה 1

1. יהיו $\begin{matrix} f\left(x_1,x_2,x_3\right):\mathbb{R}^3\to\mathbb{R} \\ g\left(x,y,z\right):\mathbb{R}^3\to\mathbb{R}^3 \end{matrix}$ מוגדרות ע"י $g\left(x,y,z\right)=\left(x+y,2x,z\right)=\left(x_1,x_2,x_3\right)$. אזי:

$$\frac{\partial}{\partial x}f\left(g\left(x,y,z\right)\right)=\sum_i\frac{\partial f}{\partial x_i}\frac{\partial x_i}{\partial x}=\frac{\partial f}{\partial x_1}\cdot 1+\frac{\partial f}{\partial x_2}\cdot 2+\frac{\partial f}{\partial x_3}\cdot 0$$

$$\frac{\partial}{\partial x}f\left(x+y,2x,z\right)=\frac{\partial f}{\partial\left(x+y\right)}+2\frac{\partial f}{\partial\left(2x\right)}$$

2. נניח $f=f_1\left(f_2...f_n\left(x\right)\right)$. נניח כי $f_i:\mathbb{R}\to\mathbb{R}$ אזי כלל השרשרת יראה כך:

$$\frac{df}{dx}=\frac{df_1}{df_2}\cdots\frac{df_{n-1}}{df_n}\frac{df_n}{dx}$$

במידה ו-x הוא וקטור אזי נקבל

$$\nabla_x f_{n-1}=\left(\frac{\partial f_n}{\partial x}\right)^T\nabla_x f_n$$

$$\nabla_x f_{n-2}=\left(\frac{\partial f_{n-1}}{\partial f_n}\right)^T\left(\frac{\partial f_n}{\partial x}\right)^T\nabla_x f_n$$

$$\vdots$$

$$\nabla_x f=\left(\frac{\partial f_1}{\partial f_2}\right)^T\cdots\left(\frac{\partial f_{n-1}}{\partial f_n}\right)^T\left(\frac{\partial f_n}{\partial x}\right)^T\nabla_x f_n=\prod_{i=1}^{n-1}\left(\frac{\partial f_i}{\partial f_{i+1}}\right)^T\left(\frac{\partial f_n}{\partial x}\right)^T\nabla_x f_n$$

כאשר $\frac{\partial f_n}{\partial x}$ היא מטריצת היעקוביאן של f_n. המימדים של הכפל יתאימו בגלל ההרכבה של הפונקציות- הקלט בכל פעם הוא כמימד הפלט של הפונקציה הפנימית.

3. אם נוסיף פונקציית אקטיבציה h נקבל $f=h_1\left(f_1\left(h_2\left(f_2...h_n\left(f_n\left(x\right)\right)\right)\right)\right)$

כלומר כל הנגזרות החלקיות של פונקציית האקטיבציה בקלט שלה הן חלק מהמכפלה.

אם נשתמש באקטיבציה של סיגמואיד, אזי מתקיים $\frac{d}{dx}\sigma\left(x\right)=\sigma\left(x\right)\left(1-\sigma\left(x\right)\right)$. אם x גדול מידי או x קטן מידי, בהכרח נקבל 0 וכל הנגזרת תתאפס.

מצד שני $\frac{d}{dx}\mathrm{ReLU}=\begin{cases}1 & x\geq 0\\ 0 & x<0\end{cases}$. לכן אם x שלילי הנגזרת שוב תתאפס, אך היתרון שכך שגם עבור ערכים מאוד גדולים של הנגזרת תשאר 1 ולא תגרום לפיצוץ בחישוב הסופי.

לכן נחפש פונקציה שאינה מתאפסת. כהשראה מהסעיפים הקודמים ניתן להרכיב פונקציה שכאשר הנגזרת שלה מתאפסת, עדיין נשאר חלק של 1+ שיהיה ניטרלי במכפלה של כלל הנגזרות החלקיות. למשל פונקציית אקטיבציה מהצורה $h\left(x\right)=\sigma\left(x\right)+x$ זהו מעין שילוב של השניים שיאפשר אי לינאריות מצד אחד, וימנע התאפסות מצד שני. של כלל המכפלה

4. יהי $f_1\left(x, f_2\left(x, f_3\left(...f_{n-1}\left(x, f_n\left(x\right)\right)\right)\right)\right)$. אזי מתקיים:

$$\frac{df_{n-1}}{dx} = \frac{\partial f_{n-1}}{\partial x} + \frac{\partial f_{n-1}}{\partial f_n}\frac{\partial f_n}{\partial x}$$

$$\frac{df_{n-2}}{dx} = \frac{\partial f_{n-2}}{\partial x} + \frac{\partial f_{n-2}}{\partial f_{n-1}}\frac{df_{n-1}}{dx} = \frac{\partial f_{n-2}}{\partial x} + \frac{\partial f_{n-2}}{\partial f_{n-1}}\frac{\partial f_{n-1}}{\partial x} + \frac{\partial f_{n-2}}{\partial f_{n-1}}\frac{\partial f_{n-1}}{\partial f_n}\frac{\partial f_n}{\partial x}$$

$$\vdots$$

$$\frac{df_1}{dx} = \frac{\partial f_1}{\partial x} + \frac{\partial f_1}{\partial f_2}\frac{\partial f_2}{\partial x} + \frac{\partial f_1}{\partial f_2}\frac{\partial f_2}{\partial f_3}\frac{\partial f_3}{\partial x}$$

$$\frac{df_1}{dx} = \frac{\partial f_1}{\partial x} + \sum_{i=2}^{n}\left(\prod_{j=2}^{i}\frac{\partial f_{j-1}}{\partial f_j}\right)\frac{\partial f_i}{\partial x}$$

מכפלת הנגזרות החלקיות במקרה זה מזכירה בדיוק את ההרכבה בסעיף הבא. כלומר גם אם הנגזרת החלקית מתאפסת, היא לא תתאפס גם עבור הפונקציה f_i שתוערך גם היא בעצמה ב־x ללא נגזרות חלקיות שיאפסו גורם זה.

5. skip connection מאפשר לבצע אופטימיזציה של הגרדיאנט ולהימנע מאיפוסו. זוהי דרך אלטרנטיבית ליצירת הגרדיאנטים.
נבחן את הנגזרת במקרה שבו $f\left(x + g\left(x + h\left(x\right)\right)\right)$. בכל שלב נסמן את הנגזרת החלקית ב־d_i:

$$\frac{\partial h\left(x\right)}{\partial x} = d_1$$

$$\frac{\partial g}{\partial x} = \frac{\partial g}{\partial\left(x + h\left(x\right)\right)}\frac{\partial\left(x + h\left(x\right)\right)}{\partial x} = \underbrace{\frac{\partial g}{\partial\left(x + h\left(x\right)\right)}}_{d_2}\left(\mathbf{1} + d_1\right) = d_2 + d_1 d_2$$

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

$$= \underbrace{\left(\frac{\partial f}{\partial\left(x + g\left(x + h\left(x\right)\right)\right)}\right)}_{d_3}\left(\mathbf{1} + d_2 + d_1 d_2\right) = \boldsymbol{d_3} + d_2 d_3 + d_1 d_2 d_3$$

כלומר הגרדיאנט בשכבה הראשונה לא יתאפס אם הקודמים לו התאפסו, לעומת הרכבה כללית בה האיבר המתקבל הוא $\frac{df}{dg}\frac{dg}{dh}\frac{dh}{dx}$. תמיד יהיה את הגורם 1 שיאפשר את השפעת הנגזרת החלקית של השכבה הנוכחית למרות השכבות האחרות.
כלומר השכבה הזו מאפשר לשרשר נגזרת חלקית שהיא כמעט הזהות.

חלק 2־ חלק תאורטי־ שאלה 2

1. תרגום אודיו לטקסט

תחילה נניח כי יש עיבוד מקדים של קטע האודיו. לכל חלון באודיו נייצר ספקטוגרמה מנורמלת, ורצף הספקטוגרמות המתקבל יהיה הקלט לרשת.
צעד זה חשוב כדי להקטין את מימד הקלט ואלפשר שימוש בLSTM ־ מילים שונות שנאמרו בזמנים שונים בעצמה שונה יתורגמו לספקטוגרמה שתציג את התדירויות לפי הזמן ויאפשרו מבט ברור יותר לגבי התוכן שנאמר באודיו.
המטרה היא שהרשת תפלוט רצף של מילים שנאמרו באודיו. נניח קיומו של מילון עבור המילים בשפה המושמעת באודיו.
הרשת תהיה many to many RNN ותכיל את החלקים הבאים:

- הקלט יועבר ברכיב bidirectional LSTM ־ זה יאפשר השפעה הן של הצלילים הקרובים בעתיד והן של הצלילים המושמעים עד כה־ כלומר חלון סביב האות בזמן, ולכן המילה המורכבת בתקווה לא תקטע באמצע.
- הפלט של הLSTM bidirectional הוא h_t בכל זמן t.
- h_tהוא קלט לשכבה FF שתמפה את h_t למרחב המילון.
- בנוסף תהיה שכבה של אקטיבציה עם softmax מיד אחרי הFF עבור כל פלט בזמן t , והמילה המתאימה ביותר מהמילון תוחזר כפלט ע"י הרשת.

הרשת עונה על האינווריאנטיות בזמן שיש בקובץ אודיו. בנוסף אות יחיד בזמן אינו מייצג מילה שלמה, ולכן מבנה LSTM המסוגל לעדכן את ההסטוריה ולאפס אותה ילמד לזהות מתי מילה מתחילה, ומתי היא מסתיימת (כפי שניתן לזהות מתי מתחיל משפט ומתי נגמר משפט) . h_t כזה יועבר דרך שכבת FF על מנת לבצע את ההטלה למרחב המילים הידוע.

2. בהינתן שאלה מתן תשובה

ראינו שפתרון בעיה זו מסתמך על עקרון מעט פשוט יותר: בהינתן שאלה ופסקה, על הרשת לקבוע היכן בפסקה הנתונה מתחילה והיכן נגמרת התשובה לשאלה.
כלומר הרשת באופן כללי היא RNN והיא בעלת מבנה BERT עם טרנספורמרים:

- הרשת המתאימה תהיה BERT כאשר הקלט לרשת הוא שרשור של השאלה, אחרי סימן מפריד, ובסוף הפסקה.
- נניח קיומו של מילון וכל מילה במשפט מיוצגת בוקטור מעל מרחב המילון.
- מיד לאחר הtransformer המרכיבים את הBERT יצורפו שתי שכבות נוספות FF: שכבה של 2 ערוצים עם אקטיבציה של softmax- כל אחד מהערוצים מציין את ההסתברות שהקלט שנראה בזמן t הוא מילה בפסקה שמתחילה את התשובה לשאלה (הערוץ הראשון) או מילה שבה נגמרת התשובה לשאלה (הערוץ השני). השכבה הבאה תחזיר 1 עבור מילים שהם חלק מהתשובה (ההסתברות בערוץ הראשון מספיק גדולה), ו0 עבור יתר המילים.

רשת מסוג זה מקבלת את השאלה והפסקה יחד- ובשלב הencoder מסוגלת ליצור קשרים משמעותיים בין כל שתי מילים בין השאלה ובין הנושא. המעבר מהשאלה לפסקה מוכל בקידוד. כפי שראינו הטרנספורמר היא מעין הרחבה לMLP שמאפשרת בחינה של קשרים בין חלקים שונים בפלט, ויצירת פלט שהוא לא קבוע באורך ומתייחס לכל אחת מהמילים בפסקה.

3. ניתוח משפט (חיובי, שלילי)

במבנה הBERT ניתן להנדס את הקלט ע"י הוספת איבר לקלט שיקרא CLS וייצג את התיוג של המשפט.

- הרשת תעבוד באופן זהה לחלוטין, אך כאשר נבחן את הפלט של הרשת נבחן את הפלט עבור הקלט CLS שמייצג את התיוג.
- הרכיב הזה של הפלט יועבר דרך שכבת FF שמחזיר ערוץ אחד ואקטיבציה של סיגמואיד- ערך גדול מחצי אומר שהמשפט חיובי וערך קטן מחצי משפט שלישי.

נבחין כי גודל הקלט אינו קבוע, אך נדרשת קלסיפיקציה של הקלט לאחת משתי מחלקות. בנוסף משום שהבעיה היא בעיה של שפה טבעית וממתקיימות בה תכונות כמו הקשרים של מילה מול מילים אחרות בקלט או שמשפטים בעלי סדר מילים שונה הם בעלי אותה משמעות- מבנה הBERT מאפשר לנצל מאפיינים אלא של עיבוד שפה טבעית באופן מיטבי.

4. סיווג תמונות למחלקות.

ע"י AlexNet:

- השכבות הראשונות הן שכבות קונבולוציה עם הרבה ערוצים, הערוצים יאפשרו להבחין בין פאטרנים שונים בתמונה. שתי השכבות האחרונות מאפשרות את הקלסיפיקציה: לאחר הקלסיפיקציה מבצעים flattening (שרשור הערכים של המטריצה לוקטור יחיד) לשכבה ונפתרים מהמשמעות המרחבית של האות.
- לבסוף מעבירים את מימד הוקטור שנוצר לוקטור כגודל מספר הקטגוריות, מבצעים softmax ומחזירים את התיוג המתאים.

  יש 2 שכבות FF כדי לחזק את יכולת הקלסיפיקציה של הרשת.

ראינו שרשתות קונבולוציה מנצלות את האינווריאנטיות במרחב של הפיקסלים בתמונה. שכבות הקונבולוציה מקודדת את התוכן של התמונה ומתרגמות מידע זה למימדים נמוכים יותר אשר מאפשרים בשכבות המאוחרות של הרשת קלסיפיקציה ותיוג ע"פ הפיצ'רים המאפיינים את התמונה.

5. תרגום של מילה בודדת

בעיית תרגום של מילה משפה אחת לשפה אחרת אינה דורשת בחינה של הקשרים בין מילים שונות. למעשה ניתן ליצור שני מילונים ומתוכם לאמן embedding לייצוג מילים כוקטורים מעל המילונים בשתי השפות: כלומר ליצור ייצוג vord2wec המאפשר קידוד של מילה לוקטור במרחב שמייצג את המשמעות והסמנטיקה של המילה ולהתנתקת מהייצוג של מילה בשפה.

- לאחר שמיפוי כזה קיים בשתי השפות מספיק ליצור רשת MLP שתקבל מילה בשפה אחת, תעביר אותה דרך שכבות FF כך שמימד וקטור הקלט ימופו למרחב ממימד וקטורים במילון המטרה.

  נניח המימד של מילון המטרה הוא d. כעת בידינו מילון עם וקטורים ממימד d של כל המילים בשפת המטרה, ווקטור של המילה שאנחנו רוצים- v- לתרגם גם הוא במימד d
- פונקציית הלוס תחושב בין הv המתקבל מהרשת לבין כל הוקטורים של המילים בשפת המטרה. המילה המתאימה היא זו שהמרחק בינה לבין הוקטור הממופה הכי קטן.
- לאחר שנמצאה המילה המתאימה, נוכל להשתמש במיפוי הקיים ולהחזיר מהוקטור המתקבל את הייצוג של מילה.

מספיק במקרה זה ללמוד פונקציה שמתארת מיפוי בין שני העולמות של המילונים.

חלק 2- חלק תאורטי- שאלה 3

עבור משימה של תרגום תמונה לטקסט:

1. ארכיטקטורה שהקלט שלה הוא משפט, והיא מייצרת תמונה על סמך המשפט. (התמונות שייכות לקבוצת מחלקות מוגדרת וניתן לקודד אותן לlatent vector ממימד נמוך)

נשתמש בארכיטקטורה pretrained BERT
הבחירה הארכיטקטורה מסוג זה משום שהיא מספקת את התכונות הבאות הנדרשות ליצירת תמונה ממשפט:

- הקלט לרשת אינו חייב להיות קבוע- variable lenght
- כל זוג מילים יכול להיות קשור יחד מבחינה נושאית וסמנטית ולמבנה של transoirmers וBERT בפרט מאפשרים לבחון קשרים בין כל המילים במשפט ללא קשר למיקומם בציר הזמן.

בארכיטקטורה זו יש את החלק שאחראי לself attention שייצור עבור כל מילה וקטור של הקשרים שלה ביחד למילים אחרות. אחרי השלב הזה נקבל וקטורים המתארים את המשפט מבחינת התוכן שלו ביחס לכל מילה.
כלומר זהו תהליך הencoding של המשפט: הקידוד של המשפט הוא הפלט שBERT מחזיר, וקטור sentence embedding.

התהליך שנדרש במשימה זו דומהל תהליך של תרגום משפט משפה אחת לשפה אחרת.

- ברגע שיצרנו קידוד למשפט בשפה אחת, נרצה לבצע לו הטלה למרחב של השפה השניה- לשם ככה נוסיף רשת MLP שתבצע את ההטלה למרחב של הוקטורים של הlatent space של התמונות.
- הלוסס הוא המרחק בין שני הוקטורים האחד הוא הפלט של הMLP, לבין הencoding של התמונה.
- על מנת לקבל את התמונה חזרה נוכל להכניס את הוקטור שהתקבל מהשלב הקודם לתוך הdecoder.

2. בהנחה שהlatent vector הוא 4 וקטורים שמתאימים ל4 רבעים של התמונה. כיצד אפשר להשתמש בשכבת attention כדי לספק תיאור מקומי על קלט של הטקסט? יש לתאר את הQ, K, V.

אם נניח כי יש לנו שכבה של 4 latent vectors, אזי הבעיה שיש לפתור כי להבין לאיזה רביע מתאים כל אחד מהוקטורים הקלט.
נבחן את הבעיה באופן הבא:
עבור 4ת הlatent vectors המתאימים ניצור !4 וקטורים כל אחד מכיל פרמוטציה אחרת של שרשור הוקטורים לוקטור אחד שמייצג את המשפט. נרצה שכל הייצוגים הבאים יהיו עוקבים עם הייצוג הזה- כלומר הוקטור שמתאים לרביע הראשון בזמן $t+1$ גם הוא יופיע בחלק הראשון של שרשור 4ת הוקטורים.
שכבת $\text{multiheadedAttention}(Q, K, V)$ עם $head_1, ..., head_{4!}$ שתאפשר למקם את הוקטורים בסדר המתאים לתמונה:

- K, V - מיפוי של !4 הפרמוטציות לעצמן- K אלו בדיוק הפרמוטציות השונות של שרשור 4ת הlatent vector
- יאותחל באופן אקראי, ובכל זמן t יתעדכן להיות וקטור הפלט של הLSTM (או כל רכיב אחר שיופיע אחרי שכבת הAttention כחלק מהRNN. הוקטור הזה מעין ייצוג של מה שיהיה התמונה בזמן τ)
- $Attention\left(Q, K, V\right) = max\ \underset{v}{\arg}\ softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V$ הוגדר בצורה זו על מנת שהatention יחזיר את ההתאמה הטובה ביותר מבין הפרמוטציות. אין משמעות ליצור סכום ממושקל בין כל הסרמוטציות שכן אנחנו מחפשים חלוקה יחידה של הוקטורים בין הרביעים ולא משקול שלהם על מני כל התמונה.

המרחק הכי קצר בין הפרמוטציה של הוקטורים לבין Q - הפלט בזמן t מייצג את הקשר בין הפלט כפי שהתקבל עד כה לבין סידור נכון של הוקטורים ביחס לרביעים.
באופן זה יהיה ביטוי לקשר המרחבי שבין סידור הוקטורים לבין הפלט שבזמן τ ישמש לdecoder לשם יצירת התמונה.

David Pitts , 313371080

Maryna Romanchuk , 323255844

IDL Ex 2

Q1

Q1.a RNN

Testing activation functions results:

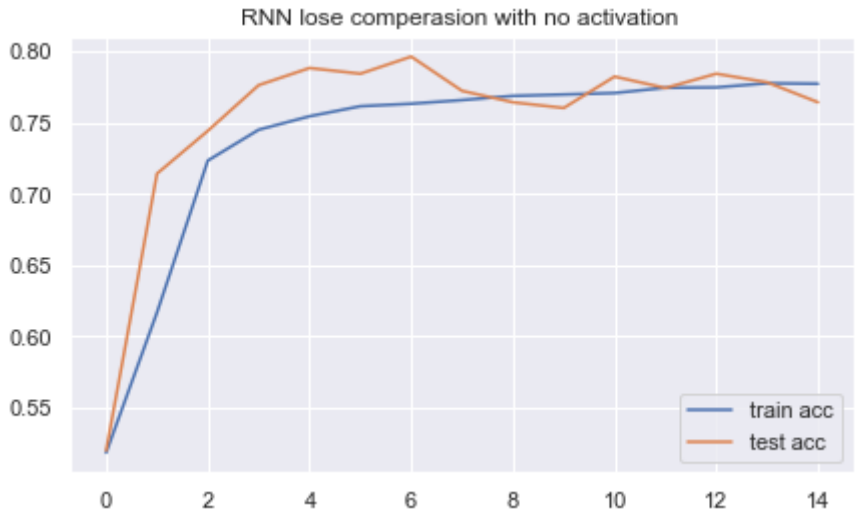
The following section will compare and attempt to explain how different activation layers impact our metrics.

Each training used the following hyper parameters:

Batch size	**128**
Epochs	**15**
Inner dimension	**64**

No activation function:

Withdrawing any activation function from the cell will result in heavily reducing the model capabilities since we take away a lot of his power to express non-linearity , but , to my surprise the results are as follows :



Compared to tanh and sigmoid accuracies I began to question my belief that adding non-linear activation function would *always* lead to better results and I understood that no activation function is better then the wrong activation function , at least under small amount of epochs.

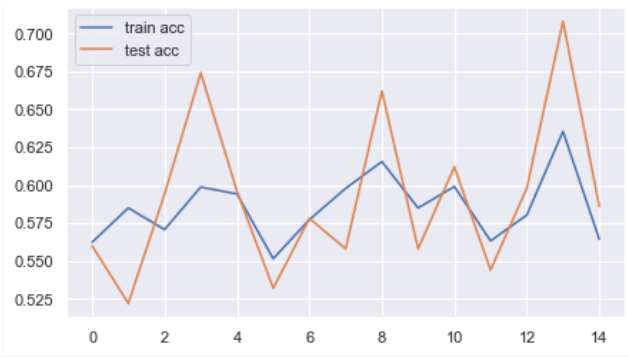
```
Accuracy score: 0.764
F1 score: 0.7457
ROC AUC score: 0.8624
```

Tanh

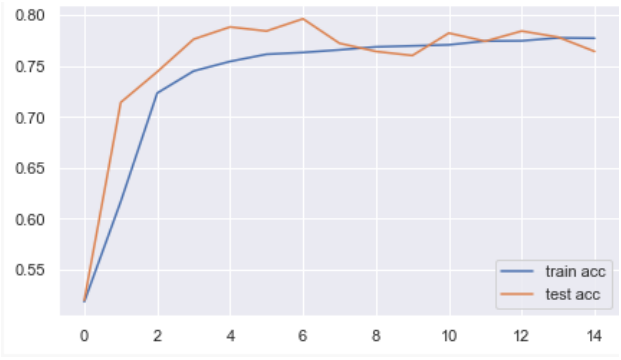
Tanh allows a reward vs punishment system , he will map negative values to negative values , and positive to positive , therefor he'll maintain the sign of the input and map it into the range (-1,1). As we can see below , training results are quite lacking and inferior in all regards to no-activation layer variation.

```
Accuracy score: 0.584
F1 score: 0.676
ROC AUC score: 0.5841
```

The graphs below declare his convergence are extremely unstable , and his accuracy is relativity low , but still better then a coin flip so some patterns are learned.



tanh activation layer train test accuracy



No activation layer train test accuracy

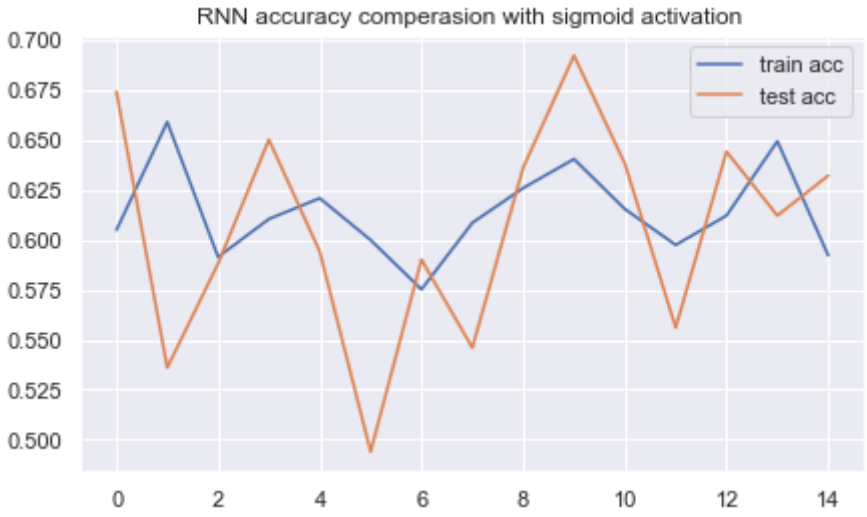
Therefor while allowing non-linearity patterns to be discovered his performance are still lacking compared to the simplest version. We came up with 2 possible explanation for it ,

- Introducing non-linearity patterns and learning them is more complicated , therefor the amount of epochs he needs to surpass non-linearity RNN's is greater , which also explain the relativity fast convergence of the former variation , since of course , less non-linear functions results almost purely a linear model which is easier to learn.
- Creating activation function doesn't fit the current tasks , we wouldn't use sigmoid for a linear regression tasks with an image in the range [-10,10] , so maybe for reason unknown to us tanh is just not the right activation function for our setting.

Sigmoid

Sigmoid maps the values from $R \rightarrow [0,1]$, therefor her behavior isn't so different from tanh and we expected her results to be lacking as well.

```
Accuracy score: 0.632
F1 score: 0.6541
ROC AUC score: 0.6961
```



To our surprise she managed to perform 5% better in accuracy then tanh , perhaps being indifferent to negative weights is the better approach to our task , but while the metrices (Beside F1) increased the problem remained , unstable learning process and lacking results.

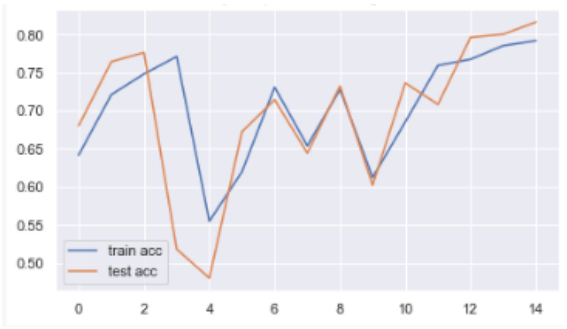
The reasons we came up with for those results are the same as before since as before , the reasons stem from the non-linearity and incorrect activation function.

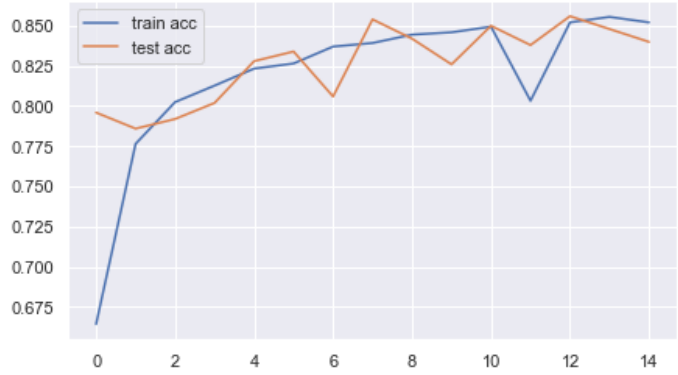
Relu

Based on our previous results we saw that functions that compress the value range to [0,1] or [-1,1] heavily hinder the results , but if we have no activation function , i.e. , not compressing the image range at all, then the results improve dramatically. Therefor we assume that ReLU will outperform the former variations since she's non-linear and doesn't compress the values.

```
Accuracy score: 0.816
F1 score: 0.8175
ROC AUC score: 0.8908
```

And as expected the results did outperform all the other variations , but like the previous non-linear activation functions, Relu convergence stability was lacking .





Unstable convergence with RELU activation function

Stable convergence with RELU activation function

Therefor we repeated the training process until a relatively stable convergence occurred , the results are comparably amazing , stable convergence improved our results by 2.4% , which indicate the the amount of variance randomness from the way the batch was shuffled and the weight initialization can impact our evaluation metrics.

```
Accuracy score: 0.84
F1 score: 0.8521
ROC AUC score: 0.9102
```

Optimizing hidden dimension

After finding the most effective activation function I trained the model 6 times , each time on a different inner-dimension and chose the highest accuracy , the table below describes those experiments and their results:

	60	**70**	**80**	**90**	**100**	**110**
accuracy	0.817	0.658	0.812	0.802	0.838	0.82

Granted , some of those results are impacted by training collapse and random seeding but we can see a tendency of increased final accuracy in average as we progress.

Another interesting fact is the training accuracy , as we increased the hidden-dimension the number of trainable parameters in the model increased , therefore our training accuracy and hidden-dimension value have a positive correlation. In other words , as we increase the number of parameters in our model the training accuracy increases *but* the test accuracy doesn't always increases as well , this is an indication of classic overfit , as the number of parameters grows the net capabilities to memorize the training data increases , which results in higher training accuracy but lower generalization capabilities.

Conclusion RNN:

Remark : We didn't check more hidden dimension values since our goal is not to optimize the hyper parameters to achieve the highest accuracies but to learn the models given to us.

The best accuracy we obtained with regard to 64 hidden-state dimensions was 84% using the relu activation function under stable convergence which resulted from repeating the training until the random initialization of the weights and the batch shuffle enabled the current net to converge in a stable manner.

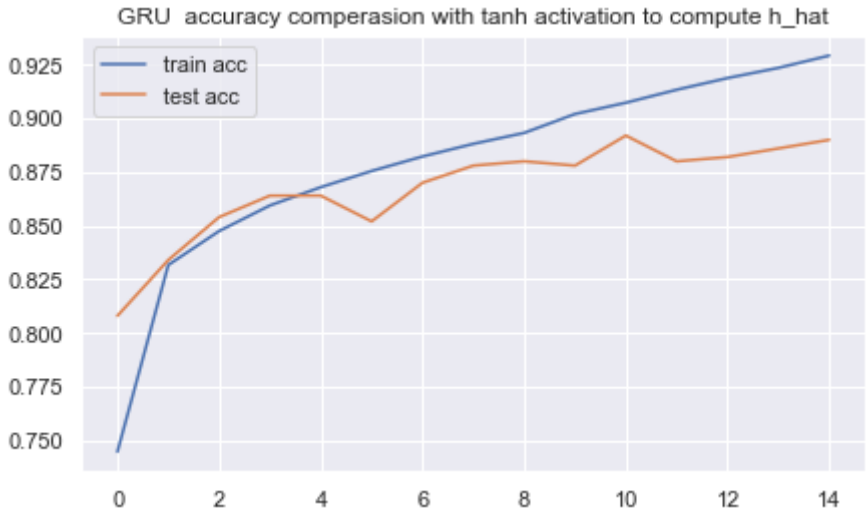
Q1.B - GRU

Activation functions:

There is less room to play with regarding the activation functions in GRU , last time we weren't bounded to gates , those gates required sigmoid in order to act as threshold for information flow. We still can attempt to see the impact of tanh on the results. GRU uses tanh in order to apply a non-linear activation over the information aggregated to output an an estimation for the history embedding at the current step (which would be interpolated with last step history based on z). As we saw previously , changing the function to Relu improved the result dramatically, so we'll briefly research the results if tanh is replaced with sigmoid , Relu and no activation function at all.

Original architecture:

Using tanh at our activation we get the following metrics :



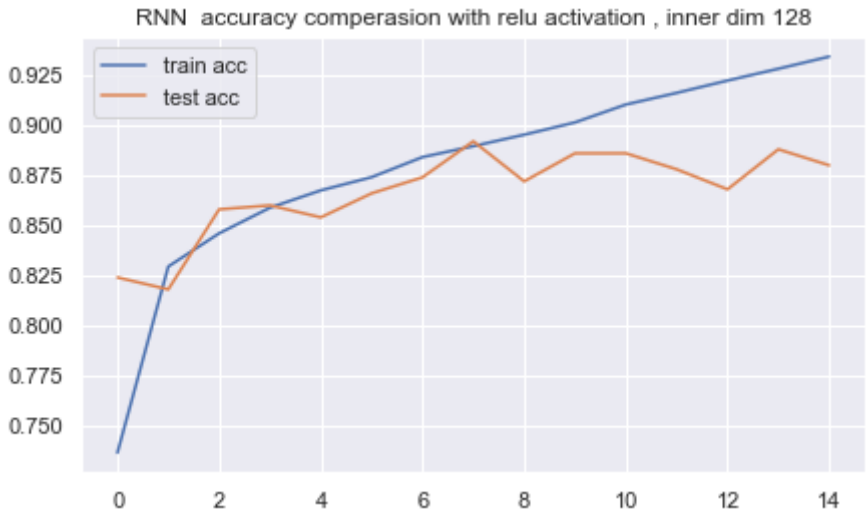

```
Accuracy score: 0.89
F1 score: 0.8964
ROC AUC score: 0.9499
```

Compared to the RNN the stable of the convergence is considerably better , even monotonic. Also the accuracy improved by 5% in the first attempt. Those results mostly derive from GRU ability to discard or encourage the parts of information he would like to preserve through his various gates which provide a more direct approach to thresholding compared to the RNN that doesn't have a built-in mechanism to discard or encourage important information.

Remark : RNN can still threshold in theory because a FC layer have the potential to map any function as we saw , but he'll have to optimize over a incredibility wider hypothesis class to find it , while the gates and thresholds of GRU allow him to reach better results faster.

No activation function:

As before , perhaps sigmoid and tanh only hurt our performance , and actually abandoning any activation completely would improve our model.




```
Accuracy score: 0.88
F1 score: 0.8885
ROC AUC score: 0.9477
```

	Predicted negative	Predicted positive
Negative	201	37
Positive	23	239

Basing my inference the previous model , the RNN's with fewer activation , and those results we can see a slight tendency toward overfitting when using fewer non-linear activation function.

Therefor non-linear activation w.r.t processing the history embedding in the GRU model outperform the former model by almost 6% accuracy , and as expected , he's worse then the original GRU model.

Sigmoid activation :

I have no intuition about what to expect from this change , I can only guess that tanh should perform better since expressing our history with negative numbers as well as positive increase our capabilities , there is no reason to limit ourselves to [0,1] if we're not interested in probabilities. The results are as follows :

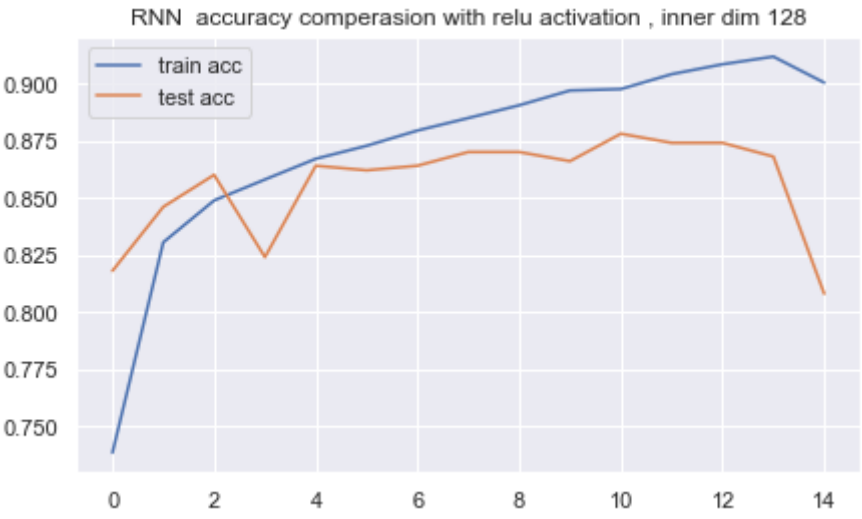

```
Accuracy score: 0.842
F1 score: 0.8512
ROC AUC score: 0.9243
```

	Predicted negative	Predicted positive
negative	195	42
positive	37	226

The results prove that sigmoid isn't the right activation layer for this task with a 5% different in accuracy from tanh.

Relu activation:

Will he repeat his prior success or fail this time? since no-activation didn't preform as good as before , I don't think ReLU would provide a dramatic improvement as before.




```
Accuracy score: 0.881
F1 score: 0.8885
ROC AUC score: 0.9477
```

	Predicted negative	Predicted positive
negative	201	37
positive	23	239

Analyzing the hidden-dimension:

	60	70	80	90	100	110
Test accuracy	0.884	0.88	0.862	0.86	0.878	0.878
Train accuracy	0.9203	0.9356	0.9477	0.9589	0.9707	0.9679

As before we can see a clear overfit of the data as we increase the hidden-dimension , therefor staying below 70 hidden-dimension would provide us with the desired results.

	20	30	40	50	60	70
Test accuracy	0.872	0.878	0.862	0.882	0.878	0.87
Train accuracy	0.8804	0.8901	0.9025	0.9138	0.9232	0.9295

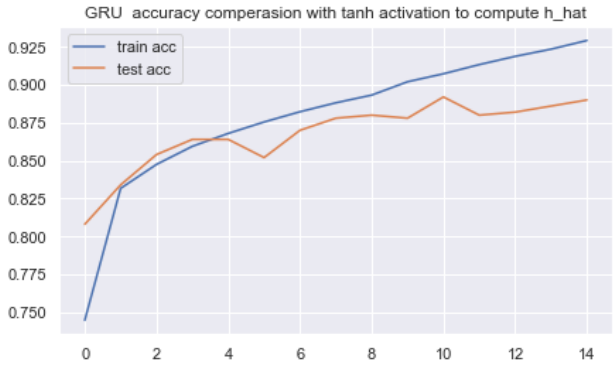
Therefor choosing a hidden dimension of 60 provides the best results with the tanh functions for GRU.

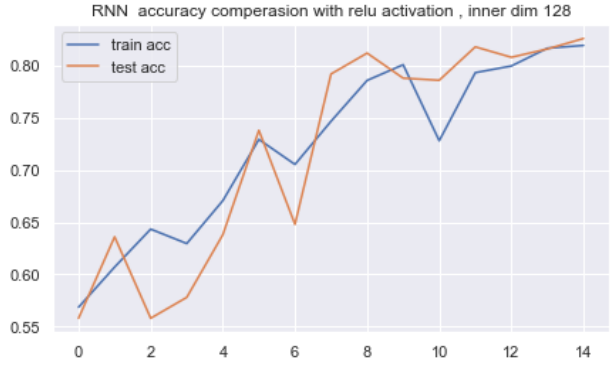
RNN vs GRU :

The dispersity in our results can also be explained by the amount of parameters the model have , RNN have 2 FC layers with an inner dimension of 64 which results in 12,673 trainable parameters , while GRU have 6 FC layers with inner dimension of 64 which results in 33,793 trainable parameters in total , more parameters increase the model capabilities. In other words , comparing between models with such a disparity parameter wise is cheating in some sense since they're not on equal footing.

To test the models capabilities on equal footing I increasing the hidden-inner dimension of RNN to 128 while keeping GRU inner-dimension to 64 would result in approximately the same amount of trainable parameters , RNN would have 33,793 parameters and GRU would have 33,793 , training both models under those setting resulted in the following :

```
Accuracy score: 0.826
F1 score: 0.8257
ROC AUC score: 0.9017
```





Accuracy score: 0.89

Accuracy score: 0.826

Therefor comparing them on equal footing regarding trainable parameters fairly reveal that GRU is a more suited model for this task , and the accuracy of the RNN even degraded when we increased his parameter size.

Q1.2

Here we are tasked with processing each word indecently in order to predict the output , therefor we ignore crossing meaning between words in the same sentence.

The followings table describes in a compact manner the process of optimizing the network hyper parameters , I chose ReLU as the activation function for the MLP and applied him once each inner-dimension to allow for high non-linearity patterns , the motivation behind choosing ReLU is because tanh and sigmoid would limit the network weights range between [-1,1] or [0,1] , there is no reason to force such a constrains on an MLP network since it'll force her to express complicated patterns under a constrained dimensions which only hurt our performance in practice even if in theory he might be able to learn it eventually.

	Layers	inner-dimension	drop out	Training accuracy	Test accuracy	Conclusion	Improvement suggestion
Ex 1	10	25	0.15	0.8	0.5880	Overfitted model	Increase dropout to 0.3

	Layers	inner-dimension	drop out	Training accuracy	Test accuracy	Conclusion	Improvement suggestion
2	10	25	0.3	0.792	0.502	Model can't generalize at all.	Reduce dropout to 0.1 and increase inner-dim to 50
3	10	50	0.1	0.853	0.8440	The model lacked capabilities to learn the complexity.	Increase inner-dimension and adjust dropout accordioning to the results.

Reaching those results based on the sample variation is sufficient I believe , below the metrics and the convergence graph is presented ,

While the train accuracy is stable , the test accuracy is increasing but still unstable , perhaps indicate an overfitting.

```
Train accuracy: 0.8533
Test accuracy: 0.8440
F1 score: 0.8465
ROC AUC score: 0.908
```

The metrics results are competitive with the simple RNN variation with RELU , which is surprising since it feels like a heuristic to handle signals , but we do have 10 trainable FC layers here compared to only 2 of the RNN , so perhaps that's what causing it.

	Predicted negative	Predicted positive
negative	207	41
positive	37	215

The confusion matrix doesn't indicate a strong favor toward one class ,i.e, a balanced training process was done.

Sentences :

As I pointed to Rannan , glove limited vocabulary would produce inaccurate match between sub_scores and test_ascii for complicated sentence , therefor the sentence below are very simple. Each sentence represent a certain difficulty NLP problems would like to over come.

1. The first sentence is strictly positive and amplification words should be relativity easy to classify.
2. The second sentence is strictly negative with amplification words.
3. The third sentence deals with negation , recognizing that "not" flip the meaning of the entire sentence is a difficult task.
4. The fourth sentence handles sarcastic sentences , frankly I don't expect our simple model to deal with it.

Positive sentence:

The following sentence includes a lot positive words , each word is weighted separately and summed , therefor his output is positive as we expected.

```
Scores for the word:  'very'  with the following score 0.1147489994764328
Scores for the word:  'good'  with the following score 0.1138630211353302
Scores for the word:  'movie'  with the following score 0.0028039589524269104
Scores for the word:  'i'  with the following score 0.08689752221107483
Scores for the word:  'love'  with the following score 0.23417869210243225
Scores for the word:  'movie'  with the following score 0.0028039589524269104
Scores for the word:  'very'  with the following score 0.1147489994764328
Scores for the word:  'love'  with the following score 0.23417869210243225
Scores for the word:  ''  with the following score 0.002768575679510832


Prediciton : Positive
Original  : Positive
```

Negative sentence:

The following sentence contains a lot of negative words therefore summing them together results in a negative output , as expected.

```
Scores for the word:  'bad'  with the following score -0.531213
Scores for the word:  'move'  with the following score -0.061157
Scores for the word:  'i'  with the following score 0.086897
Scores for the word:  'hate'  with the following score 0.002773
Scores for the word:  'it'  with the following score 0.075314
Scores for the word:  'made'  with the following score -0.064248
Scores for the word:  'me'  with the following score 0.055132
Scores for the word:  'feel'  with the following score 0.070931
Scores for the word:  'gross'  with the following score 0.051552
Scores for the word:  'and'  with the following score 0.057635
Scores for the word:  'bad'  with the following score -0.531256
Scores for the word:  'and'  with the following score 0.057631
Scores for the word:  'shallow'  with the following score -0.513632
Scores for the word:  'really'  with the following score 0.078053
Scores for the word:  'boring'  with the following score -0.78842

Prediciton : Negative
Original  : Negative
```

Negation sentence:

Therefore a positive prediction occurred but the current sentence deals with negation , since each word is considered separately we're not aware of the connectivity between the words and therefor the negation is not recognized and we predict the wrong class.

```
Scores for the word:  'not'  with the following score -0.08916131407022476
Scores for the word:  'extremely'  with the following score 0.03391373157501221
Scores for the word:  'amazing'  with the following score 0.5993659496307373
Scores for the word:  'good'  with the following score 0.1138630211353302
Scores for the word:  'love'  with the following score 0.23417869210243225
Scores for the word:  'like'  with the following score 0.0027939481660723686
Scores for the word:  'said'  with the following score 0.0028239600360393524

Prediciton : Positive
Original  : Negative
```

Sarcastic sentence:

Sarcastic sentences are very complicated structures and even a fine tuned BERT is having a hard time classifying them , almost each word have a positive sentiment in the following sentence , in order to understand the sarcasms **our model need to understand** the **disparity between the praises the person is given to the task he have completed** , this kind of analysis is **very complicated** and our model isn't capable of it in any shape or form therefor he fails.

```
Scores for the word:  'you'  with the following score 0.06675684452056885
Scores for the word:  'are'  with the following score 0.0027930457144975662
Scores for the word:  'so'  with the following score 0.002760672476142645
Scores for the word:  'capable'  with the following score -0.5330783128738403
Scores for the word:  'strong'  with the following score 0.09213486313819885
Scores for the word:  'amazing'  with the following score 0.5993659496307373
Scores for the word:  'and'  with the following score 0.057635173201560974
Scores for the word:  'good'  with the following score 0.1138630211353302
Scores for the word:  'looking'  with the following score 0.002737436443567276
Scores for the word:  'that'  with the following score 0.0028067459352314472
Scores for the word:  'even'  with the following score -0.11342062056064606
Scores for the word:  'writing'  with the following score 0.0027987370267510414
Scores for the word:  'your'  with the following score 0.02505127713084221
Scores for the word:  'own'  with the following score 0.02942352555692196
Scores for the word:  'name'  with the following score -0.05268624424934387
Scores for the word:  'is'  with the following score 0.05596131086349487
Scores for the word:  'easy'  with the following score 0.1692890226840973
Scores for the word:  'for'  with the following score 0.0028045945800840855
Scores for the word:  'you'  with the following score 0.06675684452056885
```

```
Prediciton : Positive
Original  : Negative
```

Q1.3

Weighted sum training process and metric results:

```
Accuracy score: 0.87
F1 score: 0.8752
ROC AUC score: 0.9368
```

	Predicted negative	Predicted positive
negative	207	44
positive	21	228

Training remarks :

- I used drop out and therefor the test accuracy is higher.
- The conclusion matrix indicate overfit toward positive example.

As expected the current model performed better then the former variation since we're allowed to give different importance to each word based on her semantic contribution to the sentence.

The main difference in our current variation is an addition of another mapping that's i.i.d ,i.e. ,maps each word to a constant learned importance weight that's based on her contribution to the semantic meaning of **all** the sentence in our training data , i.e. , with out dependence on the neighborhood , i.e. , i.i.d.

Therefor since the prediction over the sentence I wrote remained the same , analyzing the **different weights given to different words** would be the right way to approach it in my opinion , since there is not much to analyze other wise.

Highly impactful words:

```
Scores for the word:  'bad' the weight for the current word 0.7562177181243896
Scores for the word:  'amazing' the weight for the current word 0.8440799117088318
Scores for the word:  'good' the weight for the current word 0.24694359302520752
Scores for the word:  'easy' the weight for the current word 0.34926554560661316
Scores for the word:  'boring' the weight for the current word 1.171208381652832
Scores for the word:  'shallow' the weight for the current word 0.9326134920120239
```

Word with a miniscule impact:

```
Scores for the word:  'i' the weight for the current word -2.307689666748047
Scores for the word:  'me' the weight for the current word -3.6676599979400635
Scores for the word:  'said' the weight for the current word -5.968384742736816
Scores for the word:  'movie' the weight for the current word -1.7478996515274048
Scores for the word:  'looking' the weight for the current word -4.811641693115234
```

The main difference is that only words with semantic meaning gets a high weight , if a words appear in both negative and positive domains then she will likely balance out. An example for it can be seen by the word "movie" which is our main subject in all of our texts and therefor he should be very impactful but since he's always appearing with no distinction between good or bad , then his weight is extremely small , because while frequent he hold no semantic meaning. The same can be said about "i" or "me" that are common words to build a sentence in English and doesn't favor any kind of semantic meaning.

On the other hand every word that's charged with semantic meaning gets a very high weight , "amazing" , "bad" ,"good" ,"boring" are all words that clearly have a strong semantic meaning (ignoring negation and sarcastic sentences) and their weights are proportional it.

As I said before analyzing the sub_scores isn't really fruitful , only their order of magnitude changed because the weights changes our calculation and allow for more expression ability regarding the words. There i attached the results because it is mandatory but couldn't find anything to analyze beside the weights as I did in the previous paragraph.

Sentences:

Positive

The following sentence includes a lot positive words , each word is weighted separately and summed , therefor his output is positive as we expected.

```
Scores for the word:  'very'  with the following score 7.647510051727295
Scores for the word:  'good'  with the following score 8.508967399597168
Scores for the word:  'movie'  with the following score 2.206547737121582
Scores for the word:  'i'  with the following score 2.8893215656280518
Scores for the word:  'love'  with the following score 13.356491088867188
Scores for the word:  'movie'  with the following score 2.206547737121582
Scores for the word:  'very'  with the following score 7.647510051727295
Scores for the word:  'love'  with the following score 13.356491088867188

Prediciton : Positive
Original  : Positive
```

Negative:

The following sentence includes a lot negative words , each word is weighted separately and summed , therefor his output is positive as we expected.

```
Scores for the word:  'bad'  with the following score -13.457328796386719
Scores for the word:  'move'  with the following score 2.854335308074951
Scores for the word:  'i'  with the following score 2.8893215656280518
Scores for the word:  'hate'  with the following score 1.9233105182647705
Scores for the word:  'it'  with the following score 2.416504144668579
Scores for the word:  'made'  with the following score -6.323531150817871
Scores for the word:  'me'  with the following score 4.552388668060303
Scores for the word:  'feel'  with the following score 7.098709583282471
Scores for the word:  'gross'  with the following score 2.6295955181121826
Scores for the word:  'and'  with the following score 5.257828712463379
Scores for the word:  'bad'  with the following score -13.457328796386719
Scores for the word:  'and'  with the following score 5.257828712463379
Scores for the word:  'shallow'  with the following score -16.907142639160156
Scores for the word:  'really'  with the following score 2.0594303607940674
Scores for the word:  'boring'  with the following score -21.266456604003906

Prediciton : Negative
Original  : Negative
```

Negation

Therefore a positive prediction occurred but the current sentence deals with negation , since each word is considered separately we're not aware of the connectivity between the words and therefor the negation is not recognized and we predict the wrong class. The weighted summation we added still doesn't tackle the problem we faced before.

```
Scores for the word:  'not'  with the following score -5.10538387298584
Scores for the word:  'extremely'  with the following score 6.1918535232543945
Scores for the word:  'amazing'  with the following score 22.496076583862305
Scores for the word:  'good'  with the following score 8.508967399597168
Scores for the word:  'love'  with the following score 13.356491088867188
Scores for the word:  'like'  with the following score 2.902982473373413
Scores for the word:  'said'  with the following score 7.388331890106201


Prediciton : Positive
Original  : Negative
```

Sarcastic

Sarcastic sentences are very complicated structures and even a fine tuned BERT is having a hard time classifying them , almost each word have a positive sentiment in the following sentence , in order to understand the sarcasms **our model need to understand** the **disparity between the praises the person is given to the task he have completed** , this kind of analysis is **very complicated** and our model isn't capable of it in any shape or form therefor he fails,

in this variation of the model we added weighted summation but she doesn't solve the problem described above since it **doesn't help us to compare between the disparity of the tasks which produce the sarcastic sentence.**

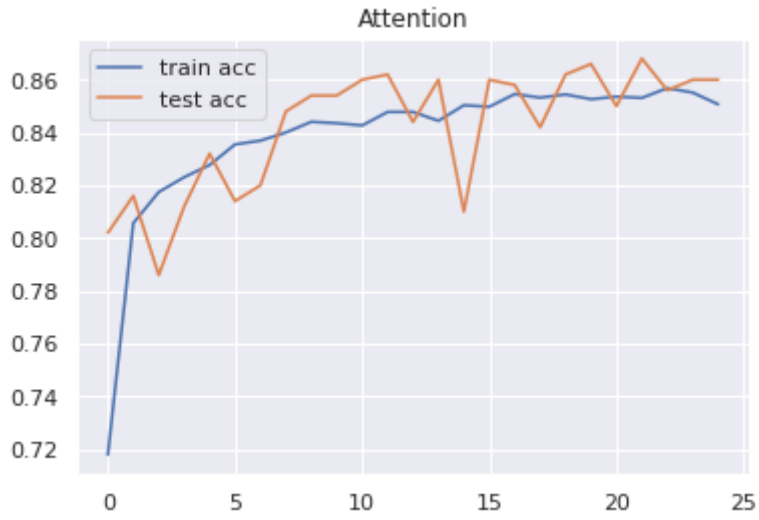
```
Scores for the word:  'you'  with the following score 2.8638572692871094
Scores for the word:  'are'  with the following score 2.239351511001587
Scores for the word:  'so'  with the following score 2.069737672805786
Scores for the word:  'capable'  with the following score 2.895043134689331
Scores for the word:  'strong'  with the following score 15.082616806030273
Scores for the word:  'amazing'  with the following score 22.496076583862305
Scores for the word:  'and'  with the following score 5.257828712463379
Scores for the word:  'good'  with the following score 8.508967399597168
Scores for the word:  'looking'  with the following score 5.944244384765625
Scores for the word:  'that'  with the following score 2.333707094192505
Scores for the word:  'even'  with the following score 0.1356588453054428
Scores for the word:  'writing'  with the following score 4.106790065765381
Scores for the word:  'your'  with the following score 1.5004218816757202
Scores for the word:  'own'  with the following score 6.642670631408691
Scores for the word:  'name'  with the following score 2.367326259613037
Scores for the word:  'is'  with the following score 2.048633337020874
Scores for the word:  'easy'  with the following score 10.744754791259766
Scores for the word:  'for'  with the following score 2.9818503856658936
Scores for the word:  'you'  with the following score 2.8638572692871094


Prediciton : Positive
Original  : Negative
```

The model still failed on the sentences with complicated logical structure , but the overall accuracy does improve since each word is weighted proportionally to her semantic contribution to the current sentence.

Attention:

Results accuracy wise :



Extremely prolonging the training and adding more layers did result in 90% accuracy but I chose to remained humble in the report since comparing the different models isn't really fair if one is given a drastic amount of resources. Also the reported accuracy is the final epoch accuracy , in a lot of training instances if we had the option to early stop the model training we could have stopped him before overfitting , the epoch at which he overfitted of course wasn't stable therefor we couldn't always get the best model.

```
Accuracy score: 0.86
```

	Predicted negative	Predicted positive
True negative	230	38
True positive	32	200

We can see a downgrade regarding accuracy since the attention model complexity can't be captured by a model as simple as before ,therefor comparing them in almost the same standards doesn't hold up.

The main principle difference in attention compared to the previous variations is the ability to cross the current word with her neighborhood , therefor the embedding the FC layers received now for each word is dependent on his neighborhood , compared to before where each word embeddings was i.i.d , i.e. , her neighborhood has no impact w.r.t the FC layer.

Positive

As expected , he managed to understand that a lot of positive words means a positive prediction , same as the previous models , I expand on the difference in those predictions after showing the examples predictions.

```
Scores for the word:  'very'  with the following score 8.24006175994873
Scores for the word:  'good'  with the following score 7.5084357261657715
Scores for the word:  'movie'  with the following score 8.032522201538086
Scores for the word:  'i'  with the following score 7.530221939086914
Scores for the word:  'love'  with the following score 11.113393783569336
Scores for the word:  'movie'  with the following score 10.475443840026855
Scores for the word:  'very'  with the following score 8.449469566345215
Scores for the word:  'love'  with the following score 10.524499893188477

Prediciton : Positive
Original  : Positive
```

Negative

As expected , he managed to understand that a lot of negative words means a positive prediction , same as the previous models , I expand on the difference in the sub values after showing the examples predictions.

```
Scores for the word:  'bad'  with the following score -8.463523864746094
Scores for the word:  'move'  with the following score 56.72036361694336
Scores for the word:  'i'  with the following score 7.530362129211426
Scores for the word:  'hate'  with the following score -3.080078363418579
Scores for the word:  'it'  with the following score 13.874954223632812
Scores for the word:  'made'  with the following score 22.942230224609375
Scores for the word:  'me'  with the following score 24.218791961669922
Scores for the word:  'feel'  with the following score 7.398444175720215
Scores for the word:  'gross'  with the following score -8.81060791015625
Scores for the word:  'and'  with the following score 7.841121673583984
Scores for the word:  'bad'  with the following score -8.447798728942871
Scores for the word:  'and'  with the following score 7.842002868652344
Scores for the word:  'shallow'  with the following score -9.450129508972168
Scores for the word:  'really'  with the following score -4.88962984085083
Scores for the word:  'boring'  with the following score -20.89261817932129


Prediciton : Negative
Original  : Negative
```

Negation

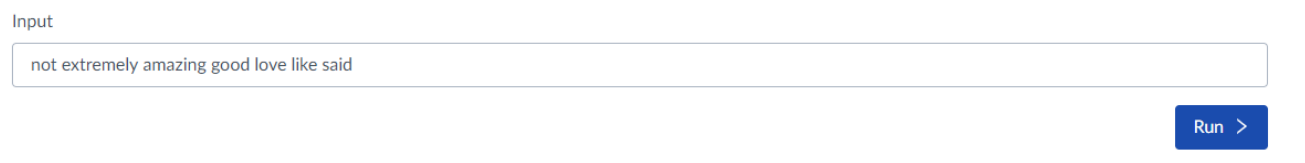
Therefore a positive prediction occurred but the current sentence deals with negation , since each word is considered separately we're not aware of the connectivity between the words and therefor the negation is not recognized and we predict the wrong class.

```
Scores for the word:  'not'  with the following score -3.0165491104125977
Scores for the word:  'extremely'  with the following score 9.75895881652832
Scores for the word:  'amazing'  with the following score 25.265445709228516
Scores for the word:  'good'  with the following score 7.478843688964844
Scores for the word:  'love'  with the following score 16.52103042602539
Scores for the word:  'like'  with the following score 54.26972579956055
Scores for the word:  'said'  with the following score 108.3665771484375


Prediciton : Positive
Original  : Negative
```

The added attention layer couldn't deal with negation ,SOTA models that are based on BERT models that are fine-tuned toward sentiment analysis handle negation pretty well , but our limited model couldn't flip the whole meaning of the sentence after seeing that the negation ,

if we go to : https://demo.allennlp.org/sentiment-analysis/MjYzMDYzMA and use the same word with RoBERTa w.r.t sentiment analysis we get the following :



Therefore usually a complicated transformer is able to connect the dots but our simplified version of self-attention couldn't.

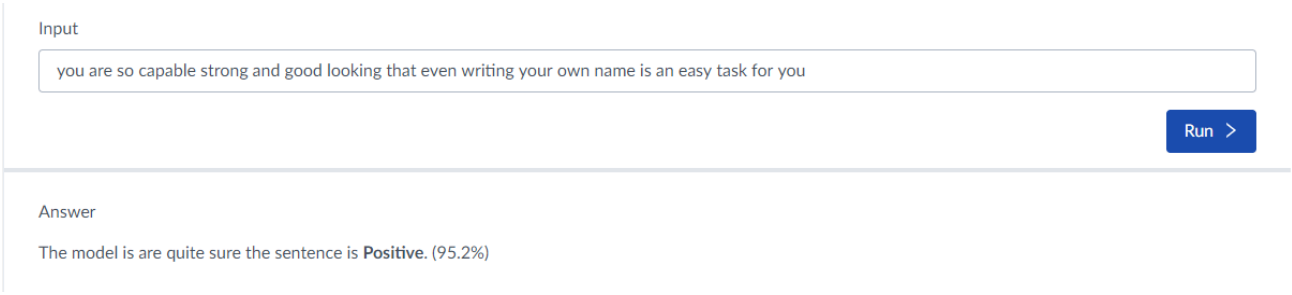
Sarcastic

Even attention can't handle the same problem as I described before , understating sarcastic sentences is a very hard problem and even if we cross those words together he can't extract the negative sentiment from any of those crosses because there is no negation at all , and there are no negative words at all. The only want to understand it is so understand the importance and disparity between the compliments and the completed tasks and our model cannot do it , therefor he regarded that sentence as any positive sentence and analyzed him in the same way.

```
Scores for the word:  'you'  with the following score 7.53505802154541
Scores for the word:  'are'  with the following score 36.3032112121582
Scores for the word:  'so'  with the following score 7.333833694458008
Scores for the word:  'capable'  with the following score 7.457080841064453
Scores for the word:  'strong'  with the following score 11.916465759277344
Scores for the word:  'amazing'  with the following score 19.900524139404297
Scores for the word:  'and'  with the following score 7.84071683883667
Scores for the word:  'good'  with the following score 7.328214168548584
Scores for the word:  'looking'  with the following score 93.61509704589844
Scores for the word:  'that'  with the following score 52.836483001708984
Scores for the word:  'even'  with the following score -4.260243892669678
Scores for the word:  'writing'  with the following score 29.740169525146484
Scores for the word:  'your'  with the following score 55.577518463134766
Scores for the word:  'own'  with the following score 41.64768981933594
Scores for the word:  'name'  with the following score 76.23111724853516
Scores for the word:  'is'  with the following score 7.535594940185547
Scores for the word:  'easy'  with the following score 7.558345317840576
Scores for the word:  'for'  with the following score 58.96645736694336
Scores for the word:  'you'  with the following score 7.521897315979004

Prediciton : Positive
Original  : Negative
```

Important to note that even SOTA models as RoBERTa are not able to deal with sarcastic sentences , therefor it is surely expected from our model as well ,as AllenNLP demo demonstrate :



Output difference :

Semantic extraction from neighbors:

In attention even if the same word appeared twice her sub_score most likely will never be the same. Comparing the positive example from Q1.3 :

```
Scores for the word:  'very'  with the following score 0.1147489994764328
Scores for the word:  'good'  with the following score 0.1138630211353302
Scores for the word:  'movie'  with the following score 0.0028039589524269104
Scores for the word:  'i'  with the following score 0.08689752221107483
Scores for the word:  'love'  with the following score 0.23417869210243225
Scores for the word:  'movie'  with the following score 0.0028039589524269104
Scores for the word:  'very'  with the following score 0.1147489994764328
Scores for the word:  'love'  with the following score 0.23417869210243225
Scores for the word:  ''  with the following score 0.002768575679510832


Prediciton : Positive
Original  : Positive
```

The word "very" have exactly the same sub score in both her appearance. But if we observe the following sub scores from the attention model

```
Scores for the word:  'very'  with the following score 8.24006175994873
Scores for the word:  'good'  with the following score 7.5084357261657715
Scores for the word:  'movie'  with the following score 8.032522201538086
Scores for the word:  'i'  with the following score 7.530221939086914
Scores for the word:  'love'  with the following score 11.113393783569336
Scores for the word:  'movie'  with the following score 10.475443840026855
Scores for the word:  'very'  with the following score 8.449469566345215
Scores for the word:  'love'  with the following score 10.524499893188477

Prediciton : Positive
Original  : Positive
```

We can see that the word "very" receive different sub scores , therefor each word mapping to her sub score is not i.i.d , i.e. , different neighborhood changes the semantic meaning and therefor the sub score of the current word. Also the model recognizes that if "love" is preceded by "I" he should get a higher score then if he's preceded by "very" , since "I love" is a declaration of the person who's saying the sentence that he's loving something , therefor charged with positive semantic , but "very love" while still positive isn't necessarily the speaker talking about himself , it could be for example "my son very love food" which is less semantically charged then "I love food".

Negative amplification

If we observe the negative example more carefully and compare the prediction we can see the following difference toward the end of the sentence :

```
Attention :

Scores for the word:  'bad'  with the following score -8.447798728942871
Scores for the word:  'and'  with the following score 7.842002868652344
Scores for the word:  'shallow'  with the following score -9.450129508972168
Scores for the word:  'really'  with the following score -4.88962984085083
Scores for the word:  'boring'  with the following score -20.89261817932129

Weighted Sum :
Scores for the word:  'bad'  with the following score -0.531274676322937
Scores for the word:  'and'  with the following score 0.057635173201560974
Scores for the word:  'shallow'  with the following score -0.5836073160171509
Scores for the word:  'really'  with the following score 0.07805226743221283
Scores for the word:  'boring'  with the following score -0.7884269952774048
```

The word "really" in the weighted sum variation got a positive score that didn't really effect the output , in other words the model was indifferent to it , the same treatment is applied to the word "and". As we know the attention models analyze his neighbor words and therefor gave gave different semantic output for them , the reason behind it stems from the fact that "and" in both variation doesn't contribute any semantic information ,the word "and" is just a continuation of a sentence in most times. On the other hand the word "really" **usually amplify something** , e.g. , "really boring" , so the model was able to understand that this **current** instance of the word "really" came to amplify the word "boring" and therefor gave her a negative sub score.

```
Weighted Sum  :
Scores for the word:  'and'  with the following score 0.057635173201560974
Scores for the word:  'really'  with the following score 0.07805226743221283

Attention:
Scores for the word:  'and'  with the following score 7.842002868652344
Scores for the word:  'really'  with the following score -4.88962984085083
```

Remark : The word really could also come in a positive sentence to amplify a positive word and will be given a positive sub score

Positive amplifications

A great example for positive amplification can be seen in the positive sentence :

```
Scores for the word:  'very'  with the following score 8.24006175994873
Scores for the word:  'good'  with the following score 7.5084357261657715
Scores for the word:  'movie'  with the following score 8.032522201538086
Scores for the word:  'i'  with the following score 7.530221939086914
Scores for the word:  'love'  with the following score 11.113393783569336
Scores for the word:  'movie'  with the following score 10.475443840026855
Scores for the word:  'very'  with the following score 8.449469566345215
Scores for the word:  'love'  with the following score 10.524499893188477
```

the word "very" appears twice , in both instances she amplify positive words (good , love). The interesting thing is that if the sub score of a given word is higher , then the amplification word ("very" in our case) will also yield higher sub score. This can be seen by comparing :

```
Scores for the word:  'very'  with the following score 8.24006175994873
Scores for the word:  'good'  with the following score 7.5084357261657715

Scores for the word:  'very'  with the following score 8.449469566345215
Scores for the word:  'love'  with the following score 10.524499893188477
```

The word "Good" sub score is lower then the word "love" sub score , and accordingly the word "very" that amplifies good gets a lower sub score then the the latter appearance of "very"

Wrong prediction :

The attention , as all other models did , failed on negation and sarcastic , those sentences logical structure is apparently too limited to learn given our extremely short training time and limited examples.

Remark : We weren't required to deeply analyze the different families of examples that act as adverbials attack to our model but another family beside Negation and Sarcasm, is "Long Dependencies" , they demonstrate the following ideas:

- "I walked with my dog to the park and it reminded me how much I love him"
- "I walked with my dog to the park and saw an orange cloud that was large and small and pretty and shallow and green and pink and brown and white and all those things just reminded me how much I like him"

Therefor we added around 30 words between the "dog " and "like" , those long dependencies would be hard to keep track of for a model and since we can always pad it as much as we'd like , we can almost always create an adversarial example to tackle our model.

Summary attention results:

Therefor we saw that attention increases that words mapping function complexity dramatically , in our current variation the previous models had the following function for each word (f maps word to subs core):

$$f(x) : \mathbb{R}^{100} \to \mathbb{R}^1 \tag{1}$$

Attention layers upgrade expression capabilities by increasing our neighborhood which in turn increases our domain but our range remained the same. Restricted attention mapping :

$$f(x) : \mathbb{R}^{1100} \to \mathbb{R}^1 \tag{2}$$

If we assume that the current words are integers values in the following range [-100,100] , we see that the previous variation have 200^{100} different vectors in her domain , but , the attention layers have 200^{1100} different vectors in her domain , and this variation is *restricted* attention , the global attention expressing ability's even more absurd. I think that quantifying the difference is the best way to demonstrate it:

$$\frac{200^{1100}}{200^{100}} = 200^{1000} \approx 2^{7000} \tag{3}$$

Which is an absurd difference in expressing ability , assuming the simple restriction I said before.

Therefor the concrete examples showed us how the the new mapping functions ,i.e. , neighborhood words affect on the final sub score , we also show that even if the mapping function became more complex our model still can't handle complex logical structures such as semantics and cynical , I do believe that Transformers in general can definitely handle those logical structures but our limited resources doesn't allow us to realize their potential.