[Improving Semantic Textual Similarity by Dividing Ambiguous Words Meanings | NLP WARRIORS](http://nlpprogress.com/english/semantic_textual_similarity.html)

הגדרת הבעיה:

שיפור ביצועים של word embedding על ידי הפרדת ייצוגים של משמעויות שונות של מילים בעלות כפל. לדוגמה למילה mouse יש שתי משמעויות – עכבר של מחשב והחיה עכבר, כאשר עושים word embedding המילה mouse מקבל ייצוג יחיד שטומן בתוכו את שתי המשמעויות האלה, מה שפוגע באיכות הייצוג. על ידי הפרדה של המילה mouse נקבל שני ייצוגים, כל אחד עבור משמעות אחת.

עיבוד מקדים:

Datasets:

מפה נוריד את הטקסט שבעזרתו נבצע clustering למילים: <http://www.nltk.org/nltk_data/>.

אלגוריתם 1

למידה (בניית המילון לשפה החדשה):

לכל מילה בטקסט הנקי שממנו לומדים את הקלאסטרים, הקלאסטרים (פיצולים) יעשה **תמיד** על טקסט נקי ולא על טקסט שפיצלנו בו מילים.

1. עבור כל מילה נמצא את כל המשפטים בהם היא מופיעה, עבור כל משפט כזה ניצור ווקטור בעזרת XXXXXXXX
2. בצע קלאסטרינג של הווקטורים ושמור לכל קלאסטר את המרכז שלו. למשל: mouse\_1: (1.84, 9.34, ……).
3. ניצור מילון ממילה נקייה למשמעויות שלה ולווקטורים שמייצגים את המשמעויות האלו.

סיווג (הפיכת משפטים משפה ישנה לשפה חדשה):

סיווג המילים למשמעויותיהן עבור הטקסט שעליו הולכים להפעיל word2vec (הפיכת טקסט נקי לטקסט עם הקלאסטרינג).

1. ניצור למשפט ווקטור בעזרת XXXXXX
2. עבור כל מילה במשפט, נבדוק במילון שייצרנו האם יש פיצול למילה זו.
   1. במידה ויש פיצול נבדוק לאיזה מן הפיצולים הווקטור של המשפט הכי קרוב ונסווג את המילה לפיצול שלה.

שימושים באלגוריתם שלנו:

בעצם איך אנחנו משפרים השוואה של משפטים בעזרת האלגוריתם שלנו.

1. אנו צריכים טקסט ענק בשפה הישנה שאותו נסווג לשפה החדשה.
2. נבנה word2vec על הטקסט בשפה החדשה.
3. עבור 2 משפטים שרוצים לבדוק דימיון בינם, נקח כל אחד מהם ונסווגם לשפה החדשה.
4. נפעיל אלגוריתם שמשתמש ב-word2vec להשוואה בין משפטים שיקבל מאיתנו את המילון שיצרנו.