# Weighted-label Word Embedding

## Internship Report

Distributed Information Systems Laboratory (**LSIR**)
École Polytechnique Fédérale de Lausanne(**EPFL**)

Written by:
Moein Khajehnejad

Supervisor:
Dr. Alex Constantin

# Preface

This report documents the work done during the internship at Distributed Information Systems Laboratory **(LSIR)**, under supervision of Dr. Alex Constantin. The report shall first give an overview of the task completed during the period of internship with technical details. Then the obtained results shall be discussed and analyzed. It will also elaborate on the the future works which can be pursued as an advancement of the current work.

# Acknowledgments

First I would like to thank Prof. Karl Aberer, for giving me the opportunity to do an internship within the organization. For me it was a valuable experience to be in Switzerland and work in his lab.
Simply put, I could not have done this job without all the help I received cheerfully from Dr. Constantin. I would specially like to thank him for proving the nice ideas to work upon.

# Introduction

The main goal of my internship was to investigate new methods to find mathematical options for adding Weights in word embedding models.
Word2vec [6] has recently been applied to a wide variety of tasks in Natural Language Processing (NLP) and Text Mining. It creates a normalized vector for each word which can only be labeled positive one or negative one. But in some cases when we want to see the effect of words being weighted like in specificity, we are looking for weighted linear combination of a group of words with real number coefficients that in the word2vec model has not been proposed.
For example we know the Top closest Concepts to "France" (by Word Embedding proximity). "France" is the only Positive label here, and it essentially has the "weight"1.0 the vector, i.e. **(France, 1.0).** We want to give each label (positive or negative) a weight when constructing the vector, for example, **(France, 1.0) + (Paris, 0.5)**.
This report presents some heuristic methods developed during this internship according to the mentioned goal above. In what follows I will try to give a brief description of these methods to determine their operation.

# Developed Methods

## *Method 1: Local Embedded Scaling (L.E.S.) :*

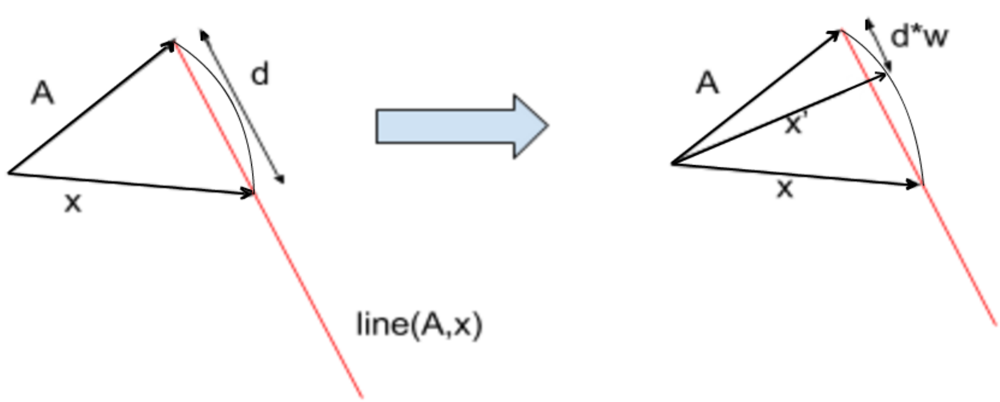
In this method we used a meaningful scaling of vectors from their original vector space to a new space.
Let us suppose we want to find the corresponding vector(or a point in the primary space) of the function "*f*" below:

$$\mathbf{f}(x_1, x_2, \ldots) = w_0 * x_0 + w_1 * x_1 + w_2 * x_2 + \ldots \qquad (1)$$

**s={$(x_i, w_i)$| $x_i$ is a specific word in vocabulary, $w_i$ is weight of $x_i$, $(x_0 = A,\ w_0 = 1)$}**

In the first place we know that all the words(vectors) are in the same space. Now we want to somehow map the words that participated in function "*f*" to a new space.
The mapped point of $A$ will be the same as the point itself in the primary space.

Now the question is how to interpret the weights of the words ?!?
To answer it, take $x_i$ as a word with weight $w_i$, the corresponding scaled vector $x_i'$ in the new space will take distance $w_i * d_i$ from vector $A$, such that $d_i$ is the distance between $A$ and $x_i$. The following picture illustrates this procedure:



Now we have to compute the function "*f*" in the new space. Since all $w_i$'s have been applied before in the mapping procedure, it is sufficient to just calculate the summation of the scaled words according to this equation : $\sum_i A + x_i'$

After solving the above equation, a new vector $v'$ will be found. We are looking for adjacent words to $v'$, but indeed we are not allowed to look for them in the new space because some of the vectors have been scaled and we do not have all of the original words. To solve this problem we need to somehow transmit vector $v'$ to the old space to find new vector $v$ .Now the adjacent vectors of $v$ are the words we were looking for. To do this transmission, we use "Local Embedding" method. In this method we assume that $v'$ is a linear combination of vectors $x_i'$ with coefficients $t_i$ that satisfy the following equations:

$$\arg\min \; v' - (t_0 * x_0' + t_1 * x_1' + t_2 * x_2' + \ldots + t_n * x_n') \quad (2)$$
$$\textbf{s.t.} \; t_0 + t_1 + t_2 + \ldots + t_n = 1$$
$$n+1 \; = \textbf{number of vectors in the new space}$$

After computing the coefficients we will use $t_i$ 's to find vector $v$ such that $v = \sum_i t_i * x_i$.

For solving equation (2) we claim that it is a "Quadratic programming". We know that the original form of "Quadratic programming" is:

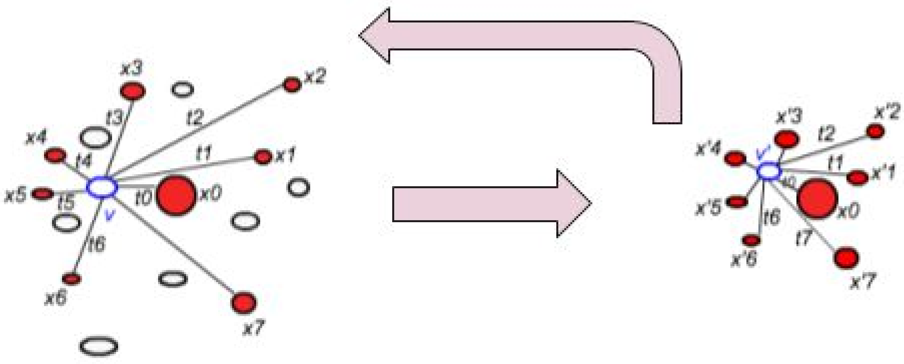
$$\textbf{minimize} \; \tfrac{1}{2} x^T Q x + c^T x \quad (3)$$
$$\textbf{Subject to} \; Ex = d$$

To prove our claim we try to rewrite equation (2) in the form of equation (3).

$$\arg\min ( v' - (t_0 * x_0' + t_1 * x_1' + t_2 * x_2' + \ldots + t_n * x_n') ) = \arg\min \; |At - v'| =$$
$$\arg\min \; |At - v'|\text{^}2 = \arg\min \; (At - v')^T (At - v') =$$
$$\arg\min ( t^T A^T A t + v'^T v' - 2 t^T A^T v' ) = \arg\min ( t^T A^T A t \; -2 (v'^T A)^T \; t \; ) =$$
$$\arg\min ( t^T Q t + 2 c^T t ) = \arg\min \; ( (1/2) \, t^T Q t + c^T t )$$

$$s.t. \; Q = A^T A \, , \; c = -A^T v'$$
$$A = \left[ x_0' \,|\, x_1' \,|\ldots|\, x_n' \right]$$
$$t = [t_0 \; t_1 \; \ldots . t_n]^T$$
$$E = [1 \; 1 \; \ldots \; 1]$$
$$d = 1$$

By solving the above “Quadratic programming”,we find vector $t$ and then we use $t$ to form vector $v$.




arg min v'- (t0*x0' + t1*x1' + t2*x2' + … + tn*xn')

s.t. t0+ t1 + t2 + … + tn = 1

n+1 = number of vectors in the new space

## Results

Below you can see the results for a few example words using this method:

Model used: ***`glove.6B.300d.bin`***

| **Sugar** | **Diabetes** | **Sugar + Diabetes (old)** | **Sugar(1.0) + Diabetes(1.0)** | **Sugar(1.0) +Diabetes(0.5)** | **Sugar(1.0) +Diabetes(0.2)** | **Diabetes(1.0) + Sugar(0.5)** | **Diabetes(1.0) +Sugar(0.2)** |
|---|---|---|---|---|---|---|---|
| | | | | | | | |
| flour 0.599 | hypertension 0.778 | hypertension 0.605 | diabetes , 0.797 | sugar , 0.948 | sugar , 0.994 | diabetes , 0.948 | diabetes , 0.994 |
| syrup 0.592 | obesity 0.722 | diabetics 0.603 | sugar , 0.797 | syrup , 0.568 | syrup , 0.591 | hypertension , 0.730 | hypertension , 0.771 |
| cocoa 0.586 | asthma 0.696 | obesity 0.577 | hypertension , 0.605 | diabetes , 0.563 | cocoa , 0.579 | obesity , 0.685 | obesity , 0.718 |
| juice 0.572 | alzheimer 0.696 | insulin 0.570 | diabetics , 0.603 | cocoa , 0.546 | flour , 0.578 | diabetics , 0.665 | asthma , 0.681 |
| corn 0.565 | arthritis 0.676 | diabetic 0.539 | obesity , 0.577 | cane , 0.545 | juice , 0.569 | asthma , 0.629 | alzheimer , 0.674 |
| cane 0.565 | diabetics 0.658 | cholesterol 0.508 | insulin , 0.570 | juice , 0.543 | cane , 0.564 | alzheimer , 0.611 | diabetics , 0.668 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| butter 0.553 | osteoporosis 0.651 | asthma 0.496 | diabetic , 0.539 | milk , 0.537 | corn , 0.559 | diabetic , 0.610 | arthritis , 0.659 |
| milk 0.553 | cardiovascular 0.625 | syrup 0.485 | cholesterol , 0.508 | corn , 0.528 | milk , 0.554 | insulin , 0.605 | osteoporosis , 0.637 |
| cotton 0.532 | disease 0.624 | glucose 0.478 | asthma , 0.496 | flour , 0.517 | butter , 0.541 | arthritis , 0.603 | diabetic , 0.623 |
| coconut 0.529 | epilepsy 0.622 | cane 0.468 | syrup , 0.485 | butter , 0.498 | cotton , 0.522 | osteoporosis , 0.586 | disease , 0.615 |
| vanilla 0.522 | cancer 0.620 | disease 0.467 | glucose , 0.478 | powdered , 0.489 | powdered , 0.516 | disease , 0.576 | cancer , 0.610 |
| powdered 0.521 | mellitus 0.618 | mellitus 0.467 | cane , 0.468 | cotton , 0.484 | coconut , 0.512 | mellitus , 0.573 | mellitus , 0.610 |
| honey 0.507 | diabetic 0.618 | arthritis 0.467 | disease , 0.467 | diabetics , 0.479 | vanilla , 0.507 | cancer , 0.570 | cardiovascular , 0.608 |
| salt 0.503 | parkinson 0.609 | milk 0.464 | mellitus , 0.467 | soy , 0.477 | honey , 0.498 | sugar , 0.563 | epilepsy , 0.604 |
| coffee 0.503 | diseases 0.596 | alzheimer 0.462 | arthritis , 0.467 | insulin , 0.477 | coffee , 0.496 | cardiovascular , 0.554 | insulin , 0.593 |
| confectioners 0.490 | rheumatoid 0.581 | cancer 0.461 | milk , 0.464 | coffee , 0.466 | salt , 0.495 | cholesterol , 0.554 | parkinson , 0.591 |
| molasses 0.487 | disorders 0.579 | osteoporosis 0.460 | alzheimer , 0.462 | sugars , 0.464 | soy , 0.486 | epilepsy , 0.549 | diseases , 0.586 |
| chocolate 0.487 | insulin 0.576 | juice 0.458 | cancer , 0.461 | coconut , 0.463 | chocolate , 0.481 | diseases , 0.547 | disorders , 0.568 |
| bananas 0.482 | dementia 0.576 | treat 0.456 | osteoporosis , 0.460 | honey , 0.463 | molasses , 0.480 | parkinson , 0.536 | rheumatoid , 0.562 |
| soy 0.482 | sclerosis 0.574 | liver 0.45 | juice , 0.458 | salt , 0.463 | confectioners , 0.48 | disorders , 0.52 | dementia , 0.55 |

Model used: ***processed_food.ALL.concepts.500d.bin***

| **Sugar/Diabetes Topic (MANUAL)** | **sugar + diabetic + diabetes + hypertension + insulin + hyperglycemia + obesity + carbohydrate + illness + blood sugar + syrup + cholesterol + diet + kidney + nutrition +** | **sugar(1) + diabetic(1) + diabetes(1) + hypertension(0.8) + insulin(0.8) + hyperglycemia(0.8) + obesity(0.8) + carbohydrate(0.69) + illness(0.6) + blood sugar(0.6) + syrup(0.6) + cholesterol(0.5) + diet(0.5) + kidney(0.4) + nutrition(0.4)** |
|---|---|---|
| | | |
| sugar, 1 | insulin_resistance 0.647 | blood_sugar , 0.758 |
| diabetic, 1 | blood_sugar_level 0.613 | blood_sugar_level , 0.691 |
| diabetes, 1 | metabolic_syndrome 0.589 | hypoglycemia , 0.643 |
| hypertension, 08 | processed_food_disease 0.580 | insulin_resistance , 0.641 |
| insulin, 08 | chronic_metabolic_disease 0.573 | sugar_level , 0.630 |
| hyperglycemia, 0.8 | risk_for_heart_disease 0.570 | insulin , 0.617 |
| obesity, 0.8 | cardiovascular_disease 0.566 | diabetes , 0.613 |
| carbohydrate, 0.69 | hypoglycemia 0.562 | processed_food_disease , 0.600 |
| illness, 0.6 | sugar_level 0.559 | diabetic , 0.594 |
| blood sugar, 0.6 | sharma_obesity 0.555 | risk_for_heart_disease , 0.585 |
| syrup, 0.6 | diabesity 0.554 | sugar , 0.578 |
| cholesterol, 0.5 | risk_of_diabetes 0.539 | metabolic_syndrome , 0.577 |
| diet, 0.5 | heart_disease 0.524 | chronic_metabolic_disease , 0.566 |
| kidney, 0.4 | risk_of_cardiovascular_disease 0.523 | hypertension , 0.562 |
| nutrition, 0.4 | high_cholesterol 0.518 | cholesterol , 0.538 |
| | triglyceride 0.515 | diabesity , 0.537 |

| | | |
|---|---|---|
| | body_ability 0.513 | body_ability , 0.533 |
| | chronic_inflammation 0.512 | cardiovascular_disease , 0.532 |
| | metabolic_obesity 0.498 | triglyceride , 0.527 |
| | low_carbohydrate_diet 0.484 | risk_of_diabetes , 0.51 |

*Note: All Word Embedding vectors in this report were generated using the models* ***`glove.6B.300d.bin`*** *and* ***`processed_food.ALL.concepts.500d.bin`***

### Pros & Cons

For large groups of words, the above results seem accurate. But in the other hand in a case with a rather small number of words, the optimization problem might not operate efficiently. Also in these cases, no matter what the overall weight of the rest of the words is, the words proximate to the one with weight one are on top of the results.

---

## Method 2: Equilibrium Point (E.P.)

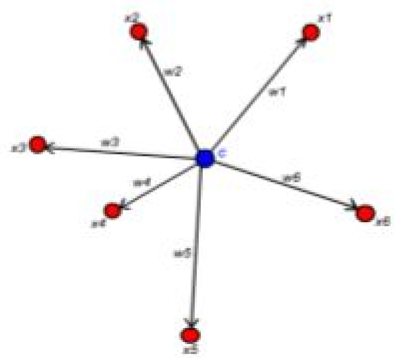
The above method works real fine. Nevertheless, in some cases it does not reach the expected results. For example suppose we want to see the results of the following summation :**(Obama,1.0) + (King,0.6) + (Queen,0.6) + (Prince,0.6)**. What we expect to see in the results is to have words that are closer to **King,Queen,Prince** on top of the ones proximate to **Obama**, because the overall effect of these similar three words in the summation is greater than of **Obama's**. However by applying the last method we will always first see the words which are closer to the elements with weight 1.0 in the results.
Now in this new method we start with computing function *"f"* to get a point $c$ as the summation result.
This question comes up here: What is the meaning of the point $c$ ?!?
To answer it, we can interpret point $c$ as the equilibrium point for the word set of function *"f"* such that the weighted vector addition of components of the below set is equal to *0* .

**s={** $\overline{cx_i}$ $\mid x_i$ **is a specific word in vocabulary,** $(x_0 = A,\ w_0 = 1)$ **}**

Due to what was mentioned above, this equation must hold :

$$w_0 * (\overline{cx_0}) \; + w_1 * (\overline{cx_1}) \; + \ldots \; + w_n * (\overline{cx_n}) \;\; = \; \sum_i w_i * (\overline{cx_i}) \;\; = 0$$

Solving the above equation gives us point $c$ as the target point of function *"f"* .




$$w_0 * (\overline{cx_0}) \; + w_1 * (\overline{cx_1}) \; + \ldots \; + w_n * (\overline{cx_n}) \;\; = \; \sum_i w_i * (\overline{cx_i}) \;\; = 0$$

## Results

You can see the results for a few example words due to this method below :

Model used: ***`glove.6B.300d.bin`***

| **Sugar** | **Diabetes** | **Sugar + Diabetes (old)** | **Sugar(1.0) + Diabetes(1.0)** | **Sugar(1.0) +Diabetes(0.5)** | **Sugar(1.0) +Diabetes(0.2)** | **Diabetes(1.0) + Sugar(0.5)** | **Diabetes(1.0) +Sugar(0.2)** |
|---|---|---|---|---|---|---|---|
| | | | | | | | |
| flour 0.599 | hypertension 0.778 | hypertension 0.605 | diabetes , 0.797 | sugar , 0.921 | sugar , 0.984 | diabetes , 0.921 | diabetes , 0.984 |
| syrup 0.592 | obesity 0.722 | diabetics 0.603 | sugar , 0.797 | diabetes , 0.625 | syrup , 0.586 | hypertension , 0.707 | hypertension , 0.761 |
| cocoa 0.586 | asthma 0.696 | obesity 0.577 | hypertension , 0.605 | syrup , 0.553 | cocoa , 0.571 | obesity , 0.666 | obesity , 0.711 |
| juice 0.572 | alzheimer 0.696 | insulin 0.570 | diabetics , 0.603 | cane , 0.532 | juice , 0.563 | diabetics , 0.657 | diabetics , 0.671 |
| corn 0.565 | arthritis 0.676 | diabetic 0.539 | obesity , 0.577 | juice , 0.528 | flour , 0.561 | sugar , 0.625 | asthma , 0.668 |
| cane 0.565 | diabetics 0.658 | cholesterol 0.508 | insulin , 0.570 | cocoa , 0.527 | cane , 0.561 | asthma , 0.603 | alzheimer , 0.657 |
| butter 0.553 | osteoporosis 0.651 | asthma 0.496 | diabetic , 0.539 | milk , 0.524 | corn , 0.551 | insulin , 0.602 | arthritis , 0.644 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| milk 0.553 | cardiovascular 0.625 | syrup 0.485 | cholesterol , 0.508 | diabetics , 0.513 | milk , 0.551 | diabetic , 0.599 | osteoporosis , 0.623 |
| cotton 0.532 | disease 0.624 | glucose 0.478 | asthma , 0.496 | corn , 0.511 | butter , 0.530 | alzheimer , 0.581 | diabetic , 0.622 |
| coconut 0.529 | epilepsy 0.622 | cane 0.468 | syrup , 0.485 | insulin , 0.504 | cotton , 0.512 | arthritis , 0.576 | disease , 0.605 |
| vanilla 0.522 | cancer 0.620 | disease 0.467 | glucose , 0.478 | flour , 0.490 | powdered , 0.509 | osteoporosis , 0.562 | mellitus , 0.601 |
| powdered 0.521 | mellitus 0.618 | mellitus 0.467 | cane , 0.468 | butter , 0.478 | coconut , 0.499 | disease , 0.555 | cancer , 0.600 |
| honey 0.507 | diabetic 0.618 | arthritis 0.467 | disease , 0.467 | powdered , 0.473 | vanilla , 0.494 | mellitus , 0.553 | insulin , 0.600 |
| salt 0.503 | parkinson 0.609 | milk 0.464 | mellitus , 0.467 | soy , 0.469 | honey , 0.489 | cancer , 0.550 | cardiovascular , 0.593 |
| coffee 0.503 | diseases 0.596 | alzheimer 0.462 | arthritis , 0.467 | hypertension , 0.467 | coffee , 0.489 | cholesterol , 0.548 | epilepsy , 0.589 |
| confectioners 0.490 | rheumatoid 0.581 | cancer 0.461 | milk , 0.464 | cotton , 0.465 | salt , 0.487 | cardiovascular , 0.528 | parkinson , 0.576 |
| molasses 0.487 | disorders 0.579 | osteoporosis 0.460 | alzheimer , 0.462 | sugars , 0.461 | soy , 0.486 | diseases , 0.527 | diseases , 0.576 |
| chocolate 0.487 | insulin 0.576 | juice 0.458 | cancer , 0.461 | obesity , 0.453 | chocolate , 0.474 | epilepsy , 0.523 | disorders , 0.558 |
| bananas 0.482 | dementia 0.576 | treat 0.456 | osteoporosis , 0.460 | coffee , 0.450 | molasses , 0.474 | parkinson , 0.511 | cholesterol , 0.556 |

Model used: `processed_food.ALL.concepts.500d.bin`

| **Sugar/Diabetes Topic (MANUAL)** | **sugar + diabetic + diabetes + hypertension + insulin + hyperglycemia + obesity + carbohydrate + illness + blood sugar + syrup + cholesterol + diet + kidney + nutrition +** | **sugar(1) + diabetic(1) + diabetes(1) + hypertension(0.8) + insulin(0.8) + hyperglycemia(0.8) + obesity(0.8) + carbohydrate(0.69) + illness(0.6) + blood sugar(0.6) + syrup(0.6) + cholesterol(0.5) + diet(0.5) + kidney(0.4) + nutrition(0.4)** |
|---|---|---|
| | | |
| sugar, 1 | insulin_resistance 0.647 | diabetes , 0.719 |
| diabetic, 1 | blood_sugar_level 0.613 | diabetic , 0.715 |
| diabetes, 1 | metabolic_syndrome 0.589 | blood_sugar , 0.677 |
| hypertension, 08 | processed_food_disease 0.580 | hypoglycemia , 0.677 |
| insulin, 08 | chronic_metabolic_disease 0.573 | insulin_resistance , 0.672 |
| hyperglycemia, 0.8 | risk_for_heart_disease 0.570 | insulin , 0.646 |
| obesity, 0.8 | cardiovascular_disease 0.566 | blood_sugar_level , 0.642 |
| carbohydrate, 0.69 | hypoglycemia 0.562 | metabolic_syndrome , 0.614 |
| illness, 0.6 | sugar_level 0.559 | processed_food_disease , 0.614 |
| blood sugar, 0.6 | sharma_obesity 0.555 | chronic_metabolic_disease , 0.604 |
| syrup, 0.6 | diabesity 0.554 | hypertension , 0.600 |
| cholesterol, 0.5 | risk_of_diabetes 0.539 | sugar_level , 0.597 |
| diet, 0.5 | heart_disease 0.524 | diabesity , 0.584 |
| kidney, 0.4 | risk_of_cardiovascular_disease 0.523 | risk_for_heart_disease , 0.571 |

| | | |
|---|---|---|
| nutrition, 0.4 | high_cholesterol 0.518 | sharma_obesity , 0.566 |
| | triglyceride 0.515 | cardiovascular_disease , 0.561 |
| | body_ability 0.513 | risk_of_diabetes , 0.554 |
| | chronic_inflammation 0.512 | body_ability , 0.529 |
| | metabolic_obesity 0.498 | high_cholesterol , 0.524 |
| | low_carbohydrate_diet 0.484 | heart_disease , 0.52 |

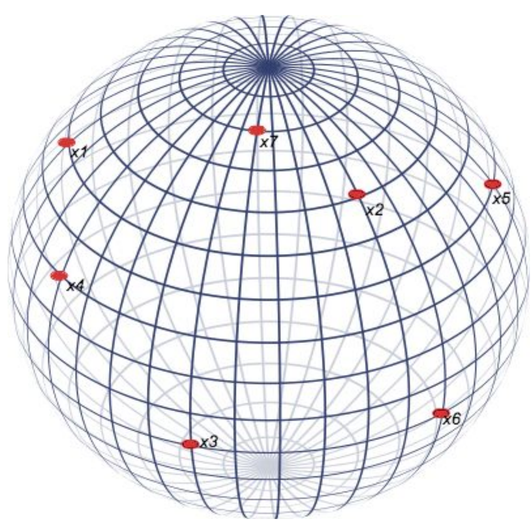
### Pros & Cons

The results obtained using this method are precise in all the cases. However although you can get enough accuracy, it is not possible to separately observe the effect of changing each word or its weight on the outcome.

---

## Method 3: Center of Gravity (C.G.)

As we know the magnitude of the vector corresponding to each word is one.Now suppose all the words lie on an n-dimensional sphere called n-sphere (hypersphere) with radius *1* and center at $O$.




The target point of function *"f"* can somehow act as the center of gravity for the set of words in the function. At first we start with not applying the weights and assuming that all $w_i$'s in function *"f"* equal one. By solving this new equation we obtain a point $c$.

Clearly there is a pulling gravitation between each word $x_i$ and point $c$ which is proportional to its weight $w_i$. As a result of these pulling forces point $c$ moves to a new location. We call it point $c'$.

$c'$ is defined as following :

$$\overline{Oc'} = \overline{Oc} + (w_0 * \overline{cx_0}) + (w_1 * \overline{cx_1}) + \ldots + (w_n * \overline{cx_n}) = \overline{Oc} + \sum_i w_i * (\overline{cx_i})$$

To find $\overline{Oc'}$ we have to compute the above vectors, using either of the two following approaches :

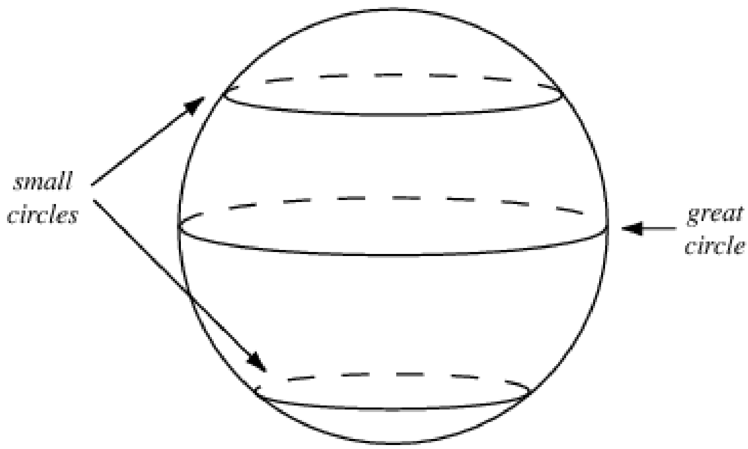
1. **Euclidean Distance**

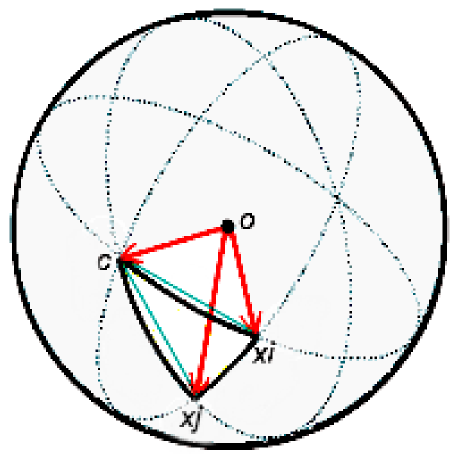
$$\overline{c\,x_i} = \overline{Oc} - \overline{O\,x_i}$$

But actually this might not look like the best approach since we are looking for the distance between vectors on the surface of the sphere, while the vector of euclidean distance connects the points inside the sphere.

2. **Spherical Distance**

The shortest path between two points on a sphere, also known as an orthodrome, is a segment of a great circle.A great circle is a section of a sphere that contains a diameter of the sphere. These segments are the desired $\overline{c\,x_i}$'s. To calculate $\overline{Oc'}$ we need to separate a fraction on each $\overline{c\,x_i}$ with magnitude of $|\overline{c\,x_i}| * w_i$ . Now we have a new set of points and their summation gives us a result which interestingly seems similar to the result obtained by the first approach.



## Results

As shown below, you can see the results for a few example words due to this method:

Model used: ***glove.6B.300d.bin***

| **Sugar** | **Diabetes** | **Sugar + Diabetes (old)** | **Sugar(1.0) + Diabetes(1.0)** | **Sugar(1.0) +Diabetes(0.5)** | **Sugar(1.0) +Diabetes(0.2 )** | **Diabetes(1.0) + Sugar(0.5)** | **Diabetes(1.0) +Sugar(0.2)** |
|---|---|---|---|---|---|---|---|
| | | | | | | | |
| flour 0.599 | hypertension 0.778 | hypertension 0.605 | diabetes , 0.797 | sugar , 0.972 | sugar , 0.997 | diabetes , 0.972 | diabetes , 0.997 |
| syrup 0.592 | obesity 0.722 | diabetics 0.603 | sugar , 0.797 | syrup , 0.580 | syrup , 0.592 | hypertension , 0.750 | hypertension , 0.774 |
| cocoa 0.586 | asthma 0.696 | obesity 0.577 | hypertension , 0.605 | cocoa , 0.562 | flour , 0.585 | obesity , 0.702 | obesity , 0.720 |
| juice 0.572 | alzheimer 0.696 | insulin 0.570 | diabetics , 0.603 | juice , 0.556 | cocoa , 0.582 | diabetics , 0.670 | asthma , 0.686 |
| corn 0.565 | arthritis 0.676 | diabetic 0.539 | obesity , 0.577 | cane , 0.556 | juice , 0.570 | asthma , 0.654 | alzheimer , 0.681 |
| cane 0.565 | diabetics 0.658 | cholesterol 0.508 | insulin , 0.570 | milk , 0.546 | cane , 0.565 | alzheimer , 0.640 | diabetics , 0.666 |
| butter 0.553 | osteoporosis 0.651 | asthma 0.496 | diabetic , 0.539 | flour , 0.545 | corn , 0.561 | arthritis , 0.629 | arthritis , 0.665 |
| milk 0.553 | cardiovascular 0.625 | syrup 0.485 | cholesterol , 0.508 | corn , 0.543 | milk , 0.554 | diabetic , 0.619 | osteoporosis , 0.642 |
| cotton 0.532 | disease 0.624 | glucose 0.478 | asthma , 0.496 | butter , 0.518 | butter , 0.545 | osteoporosis , 0.610 | diabetic , 0.622 |
| coconut 0.529 | epilepsy 0.622 | cane 0.468 | syrup , 0.485 | powdered , 0.502 | cotton , 0.526 | insulin , 0.603 | disease , 0.618 |
| vanilla 0.522 | cancer 0.620 | disease 0.467 | glucose , 0.478 | cotton , 0.502 | powdered , 0.518 | disease , 0.595 | cancer , 0.614 |
| powdered 0.521 | mellitus 0.618 | mellitus 0.467 | cane , 0.468 | diabetes , 0.490 | coconut , 0.517 | mellitus , 0.591 | cardiovascular , 0.614 |
| honey 0.507 | diabetic 0.618 | arthritis 0.467 | disease , 0.467 | coconut , 0.485 | vanilla , 0.512 | cancer , 0.590 | mellitus , 0.613 |
| salt 0.503 | parkinson 0.609 | milk 0.464 | mellitus , 0.467 | soy , 0.483 | honey , 0.501 | cardiovascular , 0.579 | epilepsy , 0.610 |
| coffee 0.503 | diseases 0.596 | alzheimer 0.462 | arthritis , 0.467 | vanilla , 0.482 | coffee , 0.498 | epilepsy , 0.574 | parkinson , 0.597 |
| confectioners 0.490 | rheumatoid 0.581 | cancer 0.461 | milk , 0.464 | coffee , 0.481 | salt , 0.498 | diseases , 0.566 | diseases , 0.589 |
| molasses 0.487 | disorders 0.579 | osteoporosis 0.460 | alzheimer , 0.462 | honey , 0.479 | soy , 0.485 | parkinson , 0.561 | insulin , 0.589 |
| chocolate 0.487 | insulin 0.576 | juice 0.458 | cancer , 0.461 | salt , 0.478 | confectioners , 0.483 | cholesterol , 0.557 | disorders , 0.572 |
| bananas 0.482 | dementia 0.576 | treat 0.456 | osteoporosis , 0.460 | chocolate , 0.466 | chocolate , 0.483 | disorders , 0.548 | rheumatoid , 0.568 |
| soy 0.482 | sclerosis 0.574 | liver 0.45 | juice , 0.458 | molasses , 0.466 | molasses , 0.483 | cancers , 0.535 | dementia , 0.56 |

Model used: ***processed_food.ALL.concepts.500d.bin***

| **Sugar/Diabetes Topic (MANUAL)** | **sugar + diabetic + diabetes + hypertension + insulin + hyperglycemia + obesity + carbohydrate + illness + blood sugar + syrup + cholesterol + diet + kidney + nutrition +** | **sugar(1) + diabetic(1) + diabetes(1) + hypertension(0.8) + insulin(0.8) + hyperglycemia(0.8) + obesity(0.8) + carbohydrate(0.69) + illness(0.6) + blood sugar(0.6) + syrup(0.6) + cholesterol(0.5) + diet(0.5) + kidney(0.4) + nutrition(0.4)** |
|---|---|---|
| | | |
| sugar, 1 | insulin_resistance 0.647 | Nestlé_Minor's , 0.293 |
| diabetic, 1 | blood_sugar_level 0.613 | Growers_Direct , 0.255 |
| diabetes, 1 | metabolic_syndrome 0.589 | luxottica , 0.252 |
| hypertension, 08 | processed_food_disease 0.580 | perishable_foodstuff , 0.222 |
| insulin, 08 | chronic_metabolic_disease 0.573 | lafarge , 0.218 |
| hyperglycemia, 0.8 | risk_for_heart_disease 0.570 | geography_of_food , 0.209 |
| obesity, 0.8 | cardiovascular_disease 0.566 | high_tech_farming , 0.200 |
| carbohydrate, 0.69 | hypoglycemia 0.562 | Tofa , 0.193 |
| illness, 0.6 | sugar_level 0.559 | orphaned_food , 0.192 |
| blood sugar, 0.6 | sharma_obesity 0.555 | cosmetically_perfect_food , 0.185 |
| syrup, 0.6 | diabesity 0.554 | global_carnage , 0.183 |
| cholesterol, 0.5 | risk_of_diabetes 0.539 | Super_8 , 0.181 |
| diet, 0.5 | heart_disease 0.524 | nestlé_free_week , 0.180 |
| kidney, 0.4 | risk_of_cardiovascular_disease 0.523 | far-away_market , 0.175 |
| nutrition, 0.4 | high_cholesterol 0.518 | happier_meal , 0.173 |
| | triglyceride 0.515 | direct_extension , 0.172 |
| | body_ability 0.513 | vulnerable_moment , 0.171 |
| | chronic_inflammation 0.512 | eimt , 0.171 |
| | metabolic_obesity 0.498 | young_butcher , 0.169 |
| | low_carbohydrate_diet 0.484 | boycottwalmart , 0.16 |

## Pros & Cons

The above approach gives us some interesting words as results which can not be obtained using the other methods introduced in this report.
As it can be seen in the table, in this manner there are some outliers among the outcomes which cause a decrease in precision

## *Method 4: Maximum Weighted Similarity (M.W.S.)*

In this last method what we are trying to do is to find a vector $C$ which is the most similar word to our set of words $\{A_1, A_2, A_3, ...\}$ . It means that the summation of this vector's weighted similarity(cosine similarity) with all of the words should be maximized. Since all the vectors have magnitude one, in this case cosine similarity will be the same as their dot product.

For given vectors $A$ and $C$ their dot product equals: $A \cdot C = AC^T$

Therefore we need to find a vector $C$ which satisfies the following equations:

$$arg\ max \sum_i w_i (A_i \cdot C) = arg\ max \sum_i w_i A_i C^T$$

$$S.T.\ ||C|| = 1$$

Now we can rewrite the above equation:

$$arg\ max \sum_i w_i A_i C^T = arg\ max \sum_i q_i C^T = arg\ max \sum_i [1, 1, .., 1] Q C^T = arg\ max \sum_i Z C^T$$

$$arg\ max \sum_i Z C^T = arg\ min \sum_i - Z C^T$$

$$S.T.\ ||C|| = 1$$

$$q_i = w_i A_i$$

$$q_i = The\ i_{th}\ row\ of\ matrix\ Q$$

$$Z = [1, 1, .., 1] Q$$

Solving the below optimization problem gives us vector $C$ :

$$arg\ min \sum_i - Z C^T$$

$$S.T.\ ||C|| = 1$$

To see how the weights or vectors affect the result, we can solve the optimization problem below in which $r_1$ and $r_2$ are arbitrary numbers:

$$arg\ max \sum_i {w_i}^{r_1} (A_i \cdot C)^{r_2} = arg\ max \sum_i {w_i}^{r_1} (A_i C^T)^{r_2}$$

$$S.T.\ ||C|| = 1$$

## Results

You can see the results for a few example words due to this method below :

Model used: ***`glove.6B.300d.bin`***

| **Sugar** | **Diabetes** | **Sugar + Diabetes (old)** | **Sugar(1.0) + Diabetes(1.0)** | **Sugar(1.0) +Diabetes(0.5)** | **Sugar(1.0) +Diabetes(0.2)** | **Diabetes(1.0) + Sugar(0.5)** | **Diabetes(1.0) +Sugar(0.2)** |
|---|---|---|---|---|---|---|---|
| | | | | | | | |
| flour 0.599 | hypertension 0.778 | hypertension 0.605 | diabetes , 0.797 | sugar , 0.921 | sugar , 0.984 | diabetes , 0.921 | diabetes , 0.984 |
| syrup 0.592 | obesity 0.722 | diabetics 0.603 | sugar , 0.797 | diabetes , 0.625 | syrup , 0.586 | hypertension , 0.707 | hypertension , 0.761 |
| cocoa 0.586 | asthma 0.696 | obesity 0.577 | hypertension , 0.605 | syrup , 0.553 | cocoa , 0.571 | obesity , 0.666 | obesity , 0.711 |
| juice 0.572 | alzheimer 0.696 | insulin 0.570 | diabetics , 0.603 | cane , 0.532 | juice , 0.563 | diabetics , 0.657 | diabetics , 0.671 |
| corn 0.565 | arthritis 0.676 | diabetic 0.539 | obesity , 0.577 | juice , 0.528 | flour , 0.561 | sugar , 0.625 | asthma , 0.668 |
| cane 0.565 | diabetics 0.658 | cholesterol 0.508 | insulin , 0.570 | cocoa , 0.527 | cane , 0.561 | asthma , 0.603 | alzheimer , 0.657 |
| butter 0.553 | osteoporosis 0.651 | asthma 0.496 | diabetic , 0.539 | milk , 0.524 | corn , 0.551 | insulin , 0.602 | arthritis , 0.644 |
| milk 0.553 | cardiovascular 0.625 | syrup 0.485 | cholesterol , 0.508 | diabetics , 0.513 | milk , 0.551 | diabetic , 0.599 | osteoporosis , 0.623 |
| cotton 0.532 | disease 0.624 | glucose 0.478 | asthma , 0.496 | corn , 0.511 | butter , 0.530 | alzheimer , 0.581 | diabetic , 0.622 |
| coconut 0.529 | epilepsy 0.622 | cane 0.468 | syrup , 0.485 | insulin , 0.504 | cotton , 0.512 | arthritis , 0.576 | disease , 0.605 |
| vanilla 0.522 | cancer 0.620 | disease 0.467 | glucose , 0.478 | flour , 0.490 | powdered , 0.509 | osteoporosis , 0.562 | mellitus , 0.601 |
| powdered 0.521 | mellitus 0.618 | mellitus 0.467 | cane , 0.468 | butter , 0.478 | coconut , 0.499 | disease , 0.555 | cancer , 0.600 |
| honey 0.507 | diabetic 0.618 | arthritis 0.467 | disease , 0.467 | powdered , 0.473 | vanilla , 0.494 | mellitus , 0.553 | insulin , 0.600 |
| salt 0.503 | parkinson 0.609 | milk 0.464 | mellitus , 0.467 | soy , 0.469 | honey , 0.489 | cancer , 0.550 | cardiovascular , 0.593 |
| coffee 0.503 | diseases 0.596 | alzheimer 0.462 | arthritis , 0.467 | hypertension , 0.467 | coffee , 0.489 | cholesterol , 0.548 | epilepsy , 0.589 |
| confectioners 0.490 | rheumatoid 0.581 | cancer 0.461 | milk , 0.464 | cotton , 0.465 | salt , 0.487 | cardiovascular , 0.528 | parkinson , 0.576 |
| molasses 0.487 | disorders 0.579 | osteoporosis 0.460 | alzheimer , 0.462 | sugars , 0.461 | soy , 0.486 | diseases , 0.527 | diseases , 0.576 |

Model used: ***processed_food.ALL.concepts.500d.bin***

| **Sugar/Diabetes Topic (MANUAL)** | **sugar + diabetic + diabetes + hypertension + insulin + hyperglycemia + obesity + carbohydrate + illness + blood sugar + syrup + cholesterol + diet + kidney + nutrition +** | **sugar(1) + diabetic(1) + diabetes(1) + hypertension(0.8) + insulin(0.8) + hyperglycemia(0.8) + obesity(0.8) + carbohydrate(0.69) + illness(0.6) + blood sugar(0.6) + syrup(0.6) + cholesterol(0.5) + diet(0.5) + kidney(0.4) + nutrition(0.4)** |
|---|---|---|
| | | |
| sugar, 1 | insulin_resistance 0.647 | diabetes , 0.696 |
| diabetic, 1 | blood_sugar_level 0.613 | diabetic , 0.694 |
| diabetes, 1 | metabolic_syndrome 0.589 | blood_sugar , 0.678 |
| hypertension, 08 | processed_food_disease 0.580 | insulin_resistance , 0.668 |
| insulin, 08 | chronic_metabolic_disease 0.573 | hypoglycemia , 0.667 |
| hyperglycemia, 0.8 | risk_for_heart_disease 0.570 | insulin , 0.643 |
| obesity, 0.8 | cardiovascular_disease 0.566 | blood_sugar_level , 0.641 |
| carbohydrate, 0.69 | hypoglycemia 0.562 | metabolic_syndrome , 0.608 |
| illness, 0.6 | sugar_level 0.559 | processed_food_disease , 0.608 |
| blood sugar, 0.6 | sharma_obesity 0.555 | chronic_metabolic_disease , 0.598 |
| syrup, 0.6 | diabesity 0.554 | hypertension , 0.598 |
| cholesterol, 0.5 | risk_of_diabetes 0.539 | sugar_level , 0.591 |
| diet, 0.5 | heart_disease 0.524 | diabesity , 0.578 |
| kidney, 0.4 | risk_of_cardiovascular_disease 0.523 | risk_for_heart_disease , 0.574 |
| nutrition, 0.4 | high_cholesterol 0.518 | sharma_obesity , 0.565 |
| | triglyceride 0.515 | cardiovascular_disease , 0.562 |
| | body_ability 0.513 | risk_of_diabetes , 0.550 |
| | chronic_inflammation 0.512 | body_ability , 0.531 |

## Pros & Cons

Applying this method provided us with accurate and good results for all the examples. Regardless of the number of the words it is an efficient approach to use.
Here we have the ability to closely observe how each word and its weight affect the outcomes by changing the defined parameters $r_1$ and $r_2$.

# Summary of Pros & Cons

| Method | Pros | Cons |
|---|---|---|
| *Loca Embedded Scaling (L.E.S.)* | Both positive and negative labels allowed<br><br>Accurate results for groups of unrelated words | Results adjacent to the words with weight one always on top,regardless of the overall weight of the rest<br><br>Optimization problem not appropriate for small number of words |
| *Equilibrium Point (E.P.)* | Both positive and negative labels allowed<br><br>Accurate results | Lack of possibility to separately observe the effect of change in the weight of each different word |
| *Center of Gravity (C.G.)* | Both positive and negative labels allowed | Some outlier words among the results<br><br>All words laying on a unit n-sphere may not be an accurate assumption |
| *Maximum Weighted Similarity (M.W.S.)* | Both positive and negative labels allowed<br><br>Accurate results<br><br>Efficient for large group of words<br><br>possibility to separately observe the effect of each word and its weight by changing $r_1$* and $r_2$* in the summation | |

*** $r_1$ and $r_2$ have been defined in method 4**

# Conclusion and Future Work

In this report we presented four different methods for Weighted-Label Word Embedding, each of which provided us with significant results. The results of all the explained methods seem somewhat similar to each other. Yet, one might notice some differences. As mentioned before ***L.E.S.*** method generally works efficiently except in some special cases where ***E.P.*** method can solve the problem and give us some real fine results which seem accurate. Although the ***C.G.*** method have good results, there exist some outlier words among them that cause the method not having too much precision. This method still can be modified in the future works. And finally in the ***M.W.S.*** method we can get precise results and also we have the authority to evaluate the effect of weights and vectors by changing $r_1$ and $r_2$. Now by developing these methods we are encouraged to play with new ideas and questions that come up afterwards.
Here we give a brief explanation on some of them:

- How to generate the perfect weights for a Topic Vector :
  Such that the possible inputs are a lot of documents manually tagged as belonging to specific topics like “Diabetes”,”Dark Matter”,” Climate Change”. Now we want to find these documents' Equilibrium Point (or Center of Gravity) (assuming weight=1 for each Doc2Vecs) and then use a weighted combination of the Documents' top concepts, to get a vector that is closest to this point.

- Topic2Doc :
  We know that we have Topic2Vec and also Doc2Vec. Now the interesting thing is to do topic detection using them.

- Taxonomy identification using Word_Embeddings :
  By playing with label weights, we can make changes in the taxonomy. Using these weighted-label models we can make identifying Hyponyms easier.

# Bibliography


1. *Guthrie, D., Allison, B., Liu, W., Guthrie, L., & Wilks, Y. (2006). A closer look at skip-gram modelling. In Proceedings of the 5th international Conference on Language Resources and Evaluation (LREC-2006) (pp. 1-4).*

2. *Schwenk, H. (2007). Continuous space language models. Computer Speech & Language, 21(3), 492-518.*

3. *Mikolov, T., Kopecký, J., Burget, L., Glembek, O., & Černocký, J. H. (2009, April). Neural network based language models for highly inflective languages. In Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on (pp. 4725-4728). IEEE.*

4. *Mikolov, T., Deoras, A., Kombrink, S., Burget, L., & Cernocký, J. (2011, August). Empirical Evaluation and Combination of Advanced Language Modeling Techniques. In INTERSPEECH (No. s 1, pp. 605-608).*

5. *Mikolov, T., Yih, W. T., & Zweig, G. (2013, June). Linguistic Regularities in Continuous Space Word Representations. In HLT-NAACL (pp. 746-751).*

6. *Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.*

7. *Levy, O., & Goldberg, Y. (2014). Neural word embedding as implicit matrix factorization. In Advances in Neural Information Processing Systems (pp. 2177-2185).*

8. *Niu, L. Q., & Dai, X. Y. (2015). Topic2Vec: Learning Distributed Representations of Topics. arXiv preprint arXiv:1506.08422.*

9. *Pennington, J., Socher, R., & Manning, C. D. (2014, October). Glove: Global Vectors for Word Representation. In EMNLP (Vol. 14, pp. 1532-1543).*

10. *Rong, X. (2014). word2vec parameter learning explained. arXiv preprint arXiv:1411.2738.*

11. *Dai, A. M., Olah, C., & Le, Q. V. (2015). Document embedding with paragraph vectors. arXiv preprint arXiv:1507.07998.*

12. *Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In Advances in neural information processing systems (pp. 3111-3119).*

13. *Perozzi, B., Al-Rfou, R., & Skiena, S. (2014, August). Deepwalk: Online learning of social representations. In Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 701-710). ACM.*

14. Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., & Mei, Q. (2015, May). Line: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web* (pp. 1067-1077). International World Wide Web Conferences Steering Committee.