Temporal Random Indexing of Context Vectors Applied to Event Detection

Yashank Singh[a,], Niladri Chatterjee[b]

[a] *Department of Mathematics, IIT Delhi*
[b] *Soumitra Dutta Chair Professor in Artificial Intelligence, IIT Delhi*

Abstract

In this paper we explore new representations for encoding language data.The general method of one-hot encoding grows linearly with the size of the word corpus in space-complexity. We address this by using Random Indexing(RI) of context vectors with non-zero entries. We propose a novel RI representation where we exploit the effect imposing a probability distribution on the number of randomized entries which leads to a class of RI representations. We also propose an algorithm that is log linear in the size of word corpus to track the semantic relationship of the query word to other words for suggesting the events that are relevant to the word in question. Finally we run simulations on the novel RI representations using the proposed algorithms for tweets relevant to the word “iPhone” and present results. The RI representation is shown to be faster and space efficient as compared to BoW embeddings.

Key Words:
Indexing methods, Temporal Random Indexing, Event Detection, Data and knowledge visualization, Randomized Algorithms

1. Introduction

Event detection has been a common application of machine learning and NLP. Although much work has been done on the same using classic techniques like Word Co-occurrence Matrix and SVD to track meaning and relationship between different words to detect an event, this paper tackles the problem with a view point of temporal random indexing of context vectors of words. As in modern times, data inflow from social platforms such as blogs, twitter etc. is extremely large, classical approaches like SVD are becoming inefficient as they run in $O(mn^2)$, $m * n$ being the dimension of the matrix which in the case of word co-occurrence matrix is the number of distinct words in the corpus(m=n). Moreover it is also difficult to track word relationships through time as even if divide the corpora on the basis of time, and apply SVD on obtained Word Co-occurrence matrices, the resulting reduced matrices would not be comparable because of different dimensions, thus giving little or no idea about shift in a word’s association with other words. We address this by random indexing of context vectors, in which the vectors of constant dimension “n” are chosen for each word and “r” random non-zero entries are distributed sparsely throughout the vector. Hence the dimension of the semantic vector associated with a word remains constant in time and words relationships are easy to track. We show the RI(random indexing) space can substitute the conventional method(BoW) for representing relationship between words and how the probability of orthogonality can significantly affect accuracy of these word representations to convey meaningful information. We also comment on the choice of dimension of random vector “n” and “r” to be chosen for a data set and further develop an algorithm to suggest a list of words which could be related to an event.

1.1. Definitions and Notations

Context Vectors : These are the vectors associated with each word, specifying its context value in k-dimensional domain. In the basic word co-occurrence matrix case,these vectors can be defined on k-dimension space (where k is the total number of distinct words in word corpus) as follows for word i in the word corpus/vocabulary:

$$c_i = (0, 0, 0...1_{i^{thplace}}, ...0, 0, 0) \;\; \& \;\; c_i.c_j = 0 \forall i \neq j \tag{1}$$

Semantic Vectors : These are the vectors that are part of the word co-occurrence matrix. The row of the matrix corresponds to the semantic vector of the word i in the corpora and can be calculated by adding the context vectors of all the words in context range of the occurrences of word i as follows:

$$sv_i = \sum_{d \in C} \sum_{-m<i<m} c_i \tag{2}$$

Here C denotes the word corpus/vocabulary and d is the occurrence of word i in data set. Throughout this paper *r* denotes

Email : mt1170756@iitd.ac.in (Yashank Singh),
niladri@maths.iitd.ac.in (Niladri Chatterjee)

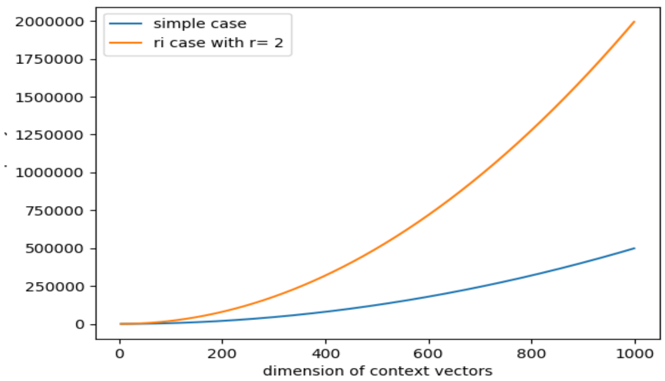

Figure 1: Word Capacity of Base Case : ${}^nC_r$ v.s RI case : ${}^nC_r \times 2^r$ r=2

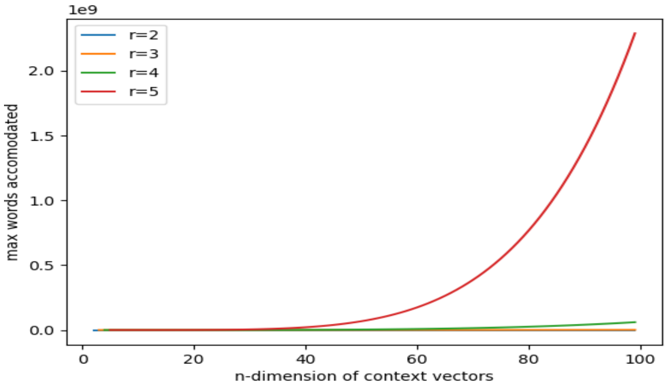

Figure 2: Word Capacity of RI case with choice between 1 and -1 for each entry : ${}^nC_r * 2^r$

the number of non-zero entries and n denotes the dimension of context vector.

2. Random Indexing

Unlike the conventional BoW representation where each word has a n dimensional basis vector e_i associated with it, where n is the size of the word corpus, in the random indexed context vectors we distribute non zero entries from a set S throughout the vector. However now the number of such context vectors that can be generated is much larger than that needed to represent each word uniquely. Hence, we aim to decrease the dimension of the context vectors through random indexing of a smaller size context vector(much smaller than number of words in corpus) by compromising on orthogonality of context vectors. The idea is to project the higher dimensional word space to a lower dimensional space spanned by randomly indexed context vector which are nearly orthogonal. Here we restrict the set S to be +1, -1. Since, we index our vectors randomly by choosing "r" non zero entries and each entry having two choices +1 and -1, a lower dimensional vector can represent more words (as r increases) to represent our data set and track relationship between different words. As if we put r=1 the context vectors are all orthogonal but the dimension increases. We establish a relation between the probability of orthogonality and accuracy of our representation to convey meaningful information. Also as size of semantic vector remains same through time, it becomes meaningful to compare semantic vectors of a word to track shift in meaning of the word with time, thus adding a temporal component to it, hence Temporal Random Indexing. The word capacity of our random indexed space with "n" as size of context vector and "r" as number of non-zero entries is shown in 1. It grows exponentially as compared to linearly in simple case.

2.1. Related Work

Random indexing based approaches for various tasks have previously been presented in QasemiZadeh and Handschuh (2015), Chatterjee and Mohan (2007), Cohen et al. (2010), Joshi et al. (2014), Sahlgren (2005), Sandin et al. (2016) QasemiZadeh (2015). In their work on Random Indexing Chatterjee and Sahoo (2013) and Chatterjee and Sahoo (2015) exploit the random indexing of context vectors for text summarization. However in their approach they distribute the +1's in the first half of the random vector and -1's in the second half sparsely. Here we do not impose any such restriction and the +1's and -1's are distributed randomly throughout the context vector. Hence this approach is novel for this task. Furthermore the idea of introducing a probability distribution on the number of non-zero entries is novel and has not been exploited in any previous work. Here we provide the comparison of our representation with the classical method(BoW) as well as that presented in Chatterjee and Sahoo (2013).Please refer to Appendix B for ablation studies. Much work has also been done exploiting the shift in the semantic vectors of words in corpus for the purpose of event detection in Jurgens and Stevens (2009), Basile et al. (2015), Basile et al. (2016). Temporal random indexing based approach for event detection in blogs proposed in Basile et al. (2015) and analysis of news data Basile et al. (2016) however fixes the number of non-zero entries to a deterministic value and hence our approach is novel, moreover no analysis w.r.t probability of orthogonality has been done in aforementioned papers. However, we study the effect of imposing a probability distribution on the number of non-zero entries in our representation.

2.2. Word Representation Capacity

The number of distinct words that can be represented is called the word representation capacity. Since, each context vector is associated with a unique word, we have that the number of context vectors must at least be the the size of the Word Corpus. Size of the BoW embeddings grows linearly with the size of the word corpus. However, as the size of RI space(number of distinct context vectors) given by

$$N(n, r, K) =^n C_r \times K^r \tag{3}$$

where n is the size of the context vector, r is the number of non-zero entries and K is the possible choices of a non-zero entry increases exponentially(as factorial is similar to exponential function), we can accommodate a large word corpus in a much smaller dimension context vector space. Also it is noteworthy

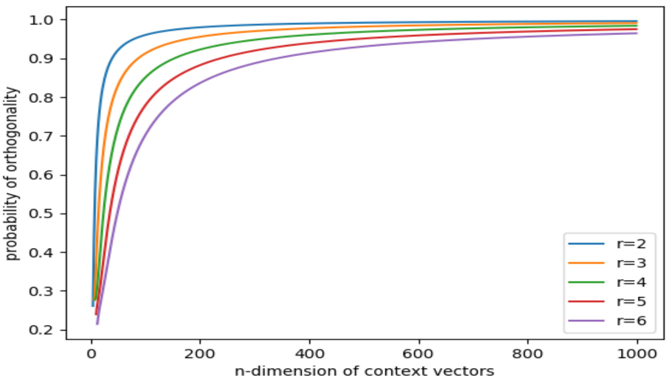

Figure 3: Probability of orthogonality of 2 vectors randomly selected in RI space where non zero entry can be 1 or -1

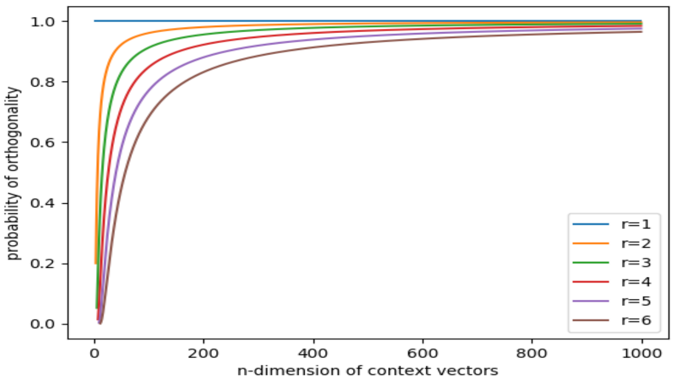

Figure 4: Probability of orthogonality of 2 vectors randomly selected in the RI space where each non zero entry is 1

to see the comparison of word capacity when the choice of each non-zero entry is “1” or one of “1”,“-1”. We see the word capacity in the latter case is much higher, which motivates to randomly index from two choices and it helps to settle for a lower dimension for a given number of words in corpus. This can be seen in Fig. 1,2.

2.3. *Probability of orthogonality*

The probability of orthogonality is the probability that two randomly selected context vectors in the n dimensional space will have the inner product as zero. As shown in Chatterjee and Sahoo (2013), that deviating from the classical representation we compromise on the probability of orthogonality. The accuracy of the representation in conveying meaningful semantic information decreases with the decrease in the probability of orthogonality(See Appendix C for more details). Hence we try to maximize the probability of orthogonality of 2 context vectors randomly selected from the RI space while decreasing the dimension of the context vectors to accurately represent semantic information and preserve word relationship information. When entries are restricted to be +1, the probability of orthogonality for n,r is given by

$$P_{ortho} = \frac{{}^{n-r}C_r}{N(n,r,1)-1} \tag{4}$$

as for a given vector the choice of orthogonal vector is the one with r non-zero entries not common to the given vector divided by the total choices given by N(n,r,1)-1. Note when r=1, it boils down to classic case and the vectors are always orthogonal to each other. However in the RI case with +1,-1, the probability can be calculated by enumeration of vectors only for small values of r. However these are readily obtained by Eqn.(8)[a more generic and rigorous treatment is provided in Section 3.] These are special cases of Eqn.(8) where $r_1 = r_2 = r$ and $P_r(r) = 1$. Here N(n,r,K) is from Eqn.(3)

$$r = 2, K = 2 : P_{ortho} = \frac{{}^{n-2}C_2 \times 2^2 + 2}{N(n,2,2)-1}$$

$$r = 3, K = 2 : P_{ortho} = \frac{{}^{n-3}C_3 \times 2^3 +^3 C_2 \times^{n-3} C_1 \times 2 \times 2}{N(n,3,2)-1}$$

$$r = 4, K = 2 : P_{ortho} = \frac{{}^{n-4}C_4 \times 2^4 +^4 C_2 \times^{n-4} C_2 \times 2^2 \times 2 + 6}{N(n,4,2)-1}$$

$$r = 5, K = 2 : P_{ortho} = \frac{{}^{n-5}C_5 \times 2^5 +^5 C_2^{n-5}C_3 \times 2^3 \times 2 +^5 C_4^{n-5}C_1 \times 2 \times 6}{N(n,5,2)-1}$$

$$r = 6, K = 2 : P_{ortho} = \frac{{}^{n-6}C_6 \times 2^6 +^6 C_2^{n-6}C_4 \times 2^4 \times 2 +^6 C_4^{n-6}C_2 \times 2^2 \times 6 + 20}{N(n,6,2)-1}$$

These are plotted in Fig.(4, 3). We observe that the probabilities of orthogonality for a given n decrease as r increases. Also, plots show that the probabilities saturate quicker(> 90%) for smaller r, and after that the increase in probability is marginal. The cut-off values of n are obtained against probability of orthogonality for the base and RI case and are shown in Table(4, 5) of Appendix A.

2.4. *Probability of orthogonality of a randomly chosen subset*

In actual practice, to represent word corpus which contains say W distinct words, we chose a subset of the set of all the vectors formed from random indexing or n dimensional context vectors and r non-zero entries. This is done to ensure a good probability of orthogonality and to account for any new context vectors that may be added to the word corpus. To motivate the choice of n and r for practical purposes the following graphs were plotted of probability of a random subset v.s the size of the RI space for a given n, r(Fig.4,5). We observe that ,initially the probability fluctuates but becomes stable after sample size reaches 200.The steady state value increases with n for a given r. These suggest that that we do not gain anything by decreasing the size of the RI-space below a threshold. The probability of orthogonality remains constant. So decreasing the size of RI-space keeping the dimension of vectors high, is shown to be non efficient, thus not usable. Hence we try to operate on the peripheral region choosing values “n” and “r” that maximise probability of orthogonality while minimising the dimension of context vectors.

3. Introducing a Probability Distribution on r

Here we provide a broader framework for random indexing based approaches. We examine the effect of introducing a probability distribution of the number of non-zero entries. We have shown that as we increase the value of r for a given n the probability of orthogonality decreases but the representation capacity measured by the max size of word corpus increases, hence this motivates the idea of a non-constant non-zero number of entries in our representation. We thus try

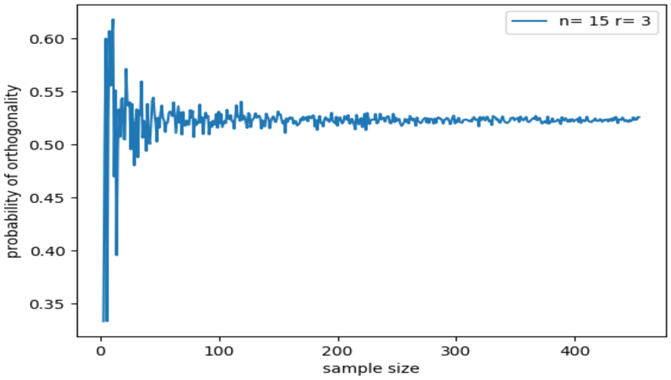

Figure 5: Probability of orthogonality of a subset for n=15, r=3

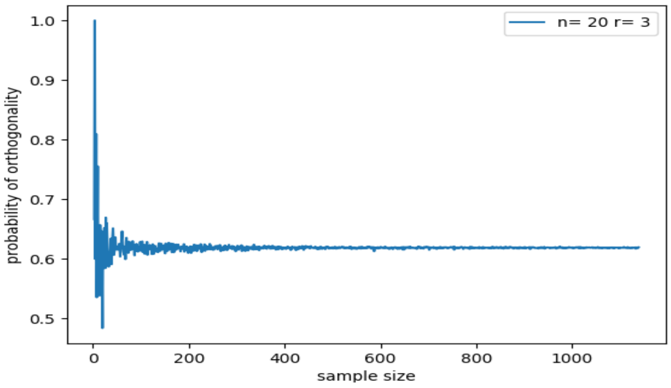

Figure 6: Probability of orthogonality of a subset for n=20, r=3

to randomize the number of non-zero entries by defining a probability distribution, such that the probability distribution takes into account that for a given n, the lesser r gives a more probability of orthogonality, but greater r gives more word space, in a dynamic case where our word space is continuously evolving, hence to constantly take care of the probability of orthogonality and also total word capacity, hence a distribution with 2 parameters n and r may be used.

3.1. Probability of orthogonality of two randomly indexed vectors of length n with r_1 and r_2 non-zero entries

We try a combinatorial approach to the problem. Let v_1 and v_2 be the vectors of length n with number of non-zero entries r_1 and r_2 respectively. WLOG, let $r2 \geq r1$ and $n \geq r_2 + r_1$, then the number of ways in which we can chose the vector with r_1 non-zero entries to make dot product zero be denoted by $\eta(r_1, r_2)$. This can be broken down in the following two cases:

Case 1 : the r_1 entries are chosen from the $n - r_2$ remaining places is

$${}^{n-r_2}C_{r_1} \times 2^{r_1}$$

Case 2 : we chose $2k$ (where $1 \leq k \leq r_1/2$ if r_1 is even , else $k \leq r_1 - 1/2$) common places out of r_2 and the remaining $r_1 - 2k$ out of $n - r_2$ places, then the number of such combinations are given by

$$\phi(2k, r_2) \times 2^{r_1 - 2k}$$

Where $\phi(2k, r_2)$ is the number of combinations where $2k$ entries are common with r_2 entries and the dot product is zero. Now since the dot product is zero, it must be an even sum of +1 and -1(as these are the only two possibilities of the product of non-zero entries). Since, there are $2k$ such products exactly k must be +1 and -1. So the problem boils down to choosing k entries out of r_2 and making the product +1 and the other k as -1. For a given vector v_2 making +1 and -1 for each entry has only 1 possibility. This gives us the following two results :

$$\phi(2k, r_2) =^{r_2} C_{2k} \times^{2k} C_k \tag{5}$$

$$\eta(r_1, r_2) = \sum_k \phi(2k, r_2) \times 2^{r_1 - 2k} \times^{n - r_2} C_{r_1 - 2k} \quad 0 \leq k \leq \lfloor r_1/2 \rfloor \tag{6}$$

Also, we have the total number of ways of choosing the vector with r_1 non-zero entries as ${}^{n}C_{r_1} \times 2^{r_1}$ which is taken care of by allowing k to be 0. Hence, the probability of orthogonality is given by :

$$P_{ortho}(r_1, r_2 | n) = \frac{\eta(r_1, r_2)}{N(n, r_1, 2)} \tag{7}$$

Note that in denominator of Eqn.(7), 1 is subtracted if $r_1 = r_2$ we chose the vectors to be distinct. This gives us a class of RI representations based on the underlying probability distribution imposed on "r". For varying discreet distributions imposed on r taking values from the set S, a number of random indexing based models can be obtained. Also note that each of these discrete distributions can simply be implemented in practice by making discretized bins in the uniform distribution over an interval and mapping them to a discreet value of r in the set S. Let the distribution on the number of non zero entries be $P_r(r = \alpha)$. $P(v_1, v_2)$ be the probability of orthogonality of two vectors v_i, v_j randomly indexed vectors with r taking discrete values from the set S, $S = \{r_1, r_2, ..., r_n\}$ where $r_i \in \mathbb{N} \forall i \geq 1$, then the expected value of the probability of orthogonality is given by :

$$\mathbb{E}(P(v_1, v_2)) = \sum_{r_i \in S, r_j \in S} P_r(r_i) \times P_r(r_j) \times P_{ortho}(r_i, r_j | n) \tag{8}$$

$P_{ortho}(r_i, r_j)$ is the probability of orthogonality given vectors of length n each having r_i and r_j non-zero entries respectively and $P_r(r_i)$ is the probability of r taking value r_i in set S. The results for deterministic case presented in Section 2.3 are degenerate cases of Eq.(8) where $r_1 = r_2 = \alpha$ and $P_r(\alpha) = 1$. Hence $\mathbb{E}(P(v_1, v_2)) = P_{ortho}(\alpha, \alpha | n)$.

3.2. Representation Capacity and Probability of orthogonality

Here we exploit an uniform distribution on the values of r in S. We plot the representation capacity and $\mathbb{E}(P(v_1, v_2))$ given by Eqn.(8) where we have uniformly distributed r in $S = \{2, 3, 4, 5, 6\}$. This is shown in 7 8. The word size runs closest to the largest word capacity i.e. the word capacity for r=6. This further motivates the idea of probabilistically distributing r. Although we gain significantly on the word capacity but we don't lose much on the probability of orthogonality by uniformly distributing r. Hence, these representations further alleviate the problem of large corpus sizes that change dynamically as these provide much space for accommodating more context vectors without compromising on the probability of orthogonality.

4. Event Detection

Event detection through sources like twitter and blogs is useful for two reasons. One being, that since these sources are not regulated, a variety of events can be detected that may otherwise may not make it to the mainstream media due to biases. Second being that these sources reflect the the event occurrence much faster than the newspapers or media houses, that take time to curate their content. Here, we use twitter data namely tweets to detect the event of launch of iPhone

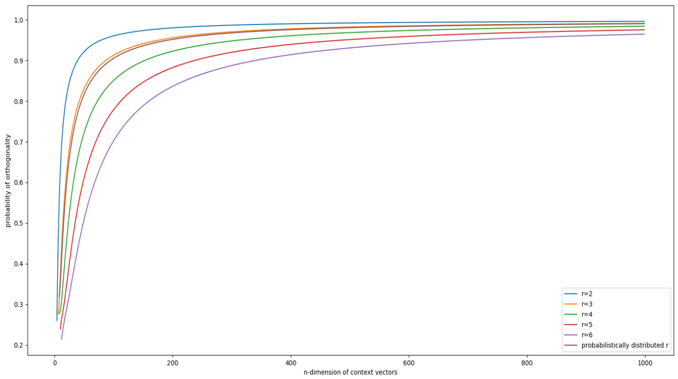

Figure 7: Expectation of probability of orthogonality

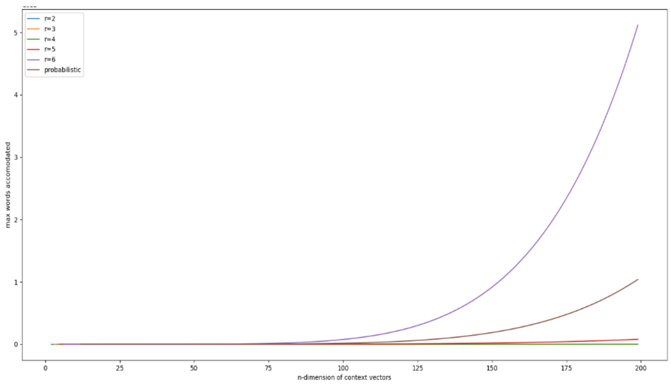

Figure 8: Expectation of word capacity

X. Since, in language data, the closely associated words usually occur close to each other, or within the context range of 5 either side of the key word, their semantic vectors are closely embedded in the n-dimensional space. Since, in time, with the change of the language data, the semantic vectors associated with these words change displaying a semantic shift that can be used to determine the changing association between our key word with respect to the other words of the corpus. Since the dimension of the context vectors and the semantic vectors remains constant throughout time in the RI bases representations, a slice of semantic vectors can be obtained spanning throughout the time period of interest and further analysis can be done to detect the occurrence of an event.

4.1. *Tracking Semantic Shift to Detect Event*

We track the semantic shift of the word vectors w.r.t the key word in question. This is possible as the length of the semantic vector remains constant in time. The value of n and r is chosen such that it is more than the size of word corpus W to ensure a good probability of orthogonality for representation of data set. As the size of the RI space grows exponentially with n, a random subset S of the total RI space was taken, such that | S| =| W| and context vectors were assigned to each distinct word in the corpora.

A bijective map is created from word space to random subset chosen $M : W \rightarrow S$. The context vectors are stored in a dictionary according to this map, so as to seamlessly add context vectors to calculate semantic vectors. For the generation of random vectors, only the non zero entries are stored for a vector in a dictionary in which each entry became a map from set $N = \{1, 2, 3, \ldots, n\}$ to A={ 1,-1} with r entries. So a context vector $\mathrm{C_i}$ for word i in word space is a map from $C_i : N^r \rightarrow A^r$ To calculate semantic vectors, the word set is simply traversed once. For the calculation of angle between semantic vectors of word i and j, functions for dot product and norm are defined and angle is calculated as :

$$\theta_{i,j} = \cos^{-1}\left(\frac{\mathbf{v_i} \cdot \mathbf{v_j}}{||\mathbf{v_i}|| \cdot ||\mathbf{v_j}||}\right) \tag{9}$$

4.2. *Algorithm for suggesting words related to event*

In this section we provide a simple approach to detect an event relating to a particular query word, by tracking the semantic shift of the words w.r.t the word in question. We do this by ranking the words according to a novel suggestion coefficient defined and compiling a list of words that are most relelvant to the event occuring concerning the key word in question.

1. Sort the list of words in the pre and post data sets according to the absolute value of the semantic angle difference measured from the word in question in ascending order.
2. Calculate normalized frequencies of the words in both the data sets(can be done while calculating semantic vectors)
3. Choose a cap of words in the post event list of words to analyse K words (where K is a hyperparameter)
4. Calculate suggestion coeff for each word in the capped list as

$$c = \Delta \times \frac{a}{b} \times r \tag{10}$$

   $\Delta = |\theta_{post} - \theta_{pre}|$ where θ is semantic angle for word i
   $\frac{a}{b}$ = ratio of ranks in the pre and post sorted lists
   r = ratio of normalized frequencies in pre and post lists
5. Sort capped list according to decreasing suggestion coeff.
6. Filter out words that were in the capped list already (add later if size of capped list $\leq p$ a hyperparameter).

The suggestion coefficient takes into account the following- a) the words whose semantic shift is larger are given more weight, b)the words those were not closer to the key word before but became closer recently are given more weight(this is done so that the words that continue to be closer are filtered out as their ranks will not change much in the sorted list) , c) that the words which were infrequent before but frequent afterwards are given more weight as those which started appearing frequently recently are more likely to be related to an event.

4.3. *Time Complexity*

Let n be the dimension of context vectors, r be the number of non-zero entries, m be the context range and $|W|$ be the size of word set. We generate $|W|$ context vectors which takes $O(r \times |W|)$ time. Then we generate $|W|$ semantic vectors which involves adding at most $2 \times m$ context vectors for each word, hence takes $O(2 \times m \times |W|)$ time. The calculation of angle for each semantic vectors takes $O(n)$ time as max number of non-zero entries can be n, and one semantic vector is traversed once for a dot product. The sorting of all the semantic vectors takes O($|W| \log |W|$) time. The sorting of the final list takes O($|k| \log |k|$) time. So overall time complexity to calculate all the angles for a given m is $O(r|W| + 2m|W| + |W| \log |W| + |k| \log |k| + n) = O(|W| \log |W|)$ as n, r, m, k are small as compared to $|W|$ and constants. Hence this runs in log linear time of word corpus. Note that when the code is run different times, the angles for RI case may be different, this is because a random subset of RI space is chosen each time as context vectors and this may differ. But the general trend in change of semantic vectors is same hence can be used for event detection.

Table 1: Suggested words related to the event concerning the query word “iPhone”

	Data Set Small	**Data Set Large**
Classical/BoW **n=$\sim$ 600,$\sim$ 1700**	*’x’, ’shoot’, ’mode’,* *’shotoniphonex’, ’iphone… ’*	*’x’, ’photo’, ’portrait’,* *’7’, ’que’*
Deterministic RI case **n=12, r=6**	*’x’, ’new’, ’iphone… ’,* *’shoot’, ’potrait’*	*’x’, ’shot’, ’puts’,* *’photography’, ’plus’*
Probabilistic RI case **r chosen randomly from** **{ 2,3,4,5,6} , n=12**	*’x’, ’plus’, ’stay’,* *’love’, ’iphonex’*	*’x’, ’shoot’, ’mode’,* *’shotoniphonex’, ’new’*

5. Implementation and Results

We chose the event “launch of iPhone X” for our analysis. The dataset comprises of tweets before the launch event and after the launch event from New York and New Delhi. The following data sets have been imported for analysis:
1.Data Set(small) : number of distinct words $\sim$ 600, imported by taking top tweets, cleaner data set.
2. Data Set(large) : number of distinct words $\sim$ 2000, imported by taking non-top tweets, random tweets also included which add noise, less cleaner data set.

5.1. Data Preprocessing

The tweets retrieved are first converted to string for further processing.

1. *Removing URLs and mentions(@username):* URLs contain the key character sequences “http” and “.com” , mentions contain “@” these are used to identify links, usernames and remove them as they do not provide any meaningful information in our case.
2. *Removing # tags and other special characters commonly used in tweets:* # tags and other characters such as ! . , “ &] [etc. are commonly used which mask the true meaningful words, these are replaced with a blank space wherever found in the string to converge to the core word.
3. *Removing STOP Words:* stop words like “me” , “you” , “is” , “am” , “they” etc. are a part of English sentences but don’t provide any meaningful observation. They are removed from data set using *nltk* library of python and importing English stop words, so as not to add these words to our final data set if they belong to the set of stop words.

5.2. Baseline for Comparison

To set a base line for tracking the shift in semantic vectors of different words, the base case is implemented as the classical case choosing orthogonal vectors of the dimension of size of word corpus.The context vectors assigned as given by 1 and semantic vectors are calculated using 2 for context(m) ranging from 1 to 10. The angle between semantic vectors of different words and “iphone” were calculated pre and post the launch.

5.3. Results

We run the algorithm for each representation for 10 times and show the average running time. Here we report the top 5 words ranked according to Eqn.10 i.e. most relevant to the event happening with a context range of 5 and time taken in milliseconds taken by each representation averaged over 10 iterations. These results are shown in Tables- 1,2. We find that the algorithm correctly predicts words such as “*x*” , “*new*” , “*portrait*” and is faster than the classical representation on an average. It is also noteworthy that distributing “r” probabilistically further gives an edge in processing time and accuracy of representation. We also see that before launch “iPhone” is closer to the word “8” as people tweet about the iPhone 8 that is the prevailing iPhone version and post launch becomes closer to “x” owing to the launch of the new version. RI representations correctly track this relationship while being significantly faster than BoW representation. This is show in Table 3.

6. Conclusion

In this paper we have shown that embeddings generated through random indexing of context vectors can be an efficient substitute for the sparse BoW embeddings used in traditional NLP tasks. The RI embeddings are shown to be computed faster than BoW embeddings and also provide an added scope of time in wake of their constant dimension. Furthermore, we provide with a novel idea of Probabilistic RI embeddings, which are shown to have larger capacity than both RI and BoW embeddings for a given dimension and can be used for online tasks where the corpus keeps changing. The ammortized time complexity of Probabilistic RI embeddings beat that of RI embeddings and sparse BoW, hence these me be further explored for online tasks. We further develop a novel algorithm to track semantic shift in relationship of words from scratch. The event detection task is closely related with tracking semantic shift in word meanings. These embeddings prove to be performing at par with the sparse BoW embeddings for the task at hand and are more efficient in computation and time as has been shown in the resutls. These embeddings can also be chosen as a better initialisation point for further learning tasks such as clustering where sparse BoW is rather expensive in computation and memory if the corpus is large in size.

References

Basile, P., Caputo, A., Semeraro, G., 2015. Temporal random indexing: A system for analysing word meaning over time. Italian Journal of Computational Linguistics 1, 55–68.

Basile, P., Caputo, A., Semeraro, G., 2016. Temporal random indexing: a tool for analysing word meaning variations in news, in: NewsIR@ECIR.

Chatterjee, N., Mohan, S., 2007. Extraction-based single-document summarization using random indexing, in: 19th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2007), IEEE. pp. 448–455.

Chatterjee, N., Sahoo, P.K., 2013. Effect of near-orthogonality on random indexing based extractive text summarization. International Journal of Innovation and Applied Studies 3, 701–713.

Chatterjee, N., Sahoo, P.K., 2015. Random indexing and modified random indexing based approach for extractive text summarization. Computer Speech & Language 29, 32–44.

Table 2: Time taken in milli Seconds for 10 iterations of algorithm for each representation

	Data Set Small	**Data Set Large**
Classical/BoW **n=∼ 600,∼ 1700**	*M=2, t =50.36 M=5, t=73.10* *M=8, t= 127.61*	*M=2, t=441.27 M=5, t=805.32* *M=8, t=993.08*
Deterministic RI **n=12, r=6**	*M=2, t=35.62 M=5, t=68.57* *M=8, t=57.52*	*M=2, t=123.50 M=5, t=191.51* *M=8, t=262.16*
Probabilistic RI case **r chosen randomly from** **{ 2,3,4,5,6} , n=12**	*M=2, t=45.87 M=5, t=44.08* *M=8, t=47.17*	*M=2, t=110.31 M=5, t=194.97* *M=8, t=222.04*

Table 3: angle between semantic vectors of words and “iPhone”

words	Data Set Small Pre launch	Data Set Small Post Launch	Data Set Large Pre Launch	Data Set Large Post Launch
Baseline				
8	0.83	1.06	0.75	0.79
x	1.28	0.83	1.28	0.72
RI case with deterministic r=6, n=12				
8	0.53	0.61	0.66	0.51
x	0.81	0.43	1.06	0.36
RI case with probabilistic r distributed uniformly in {2,3,4,5,6}				
8	0.66	0.57	0.50	0.42
x	0.94	0.47	0.77	0.33

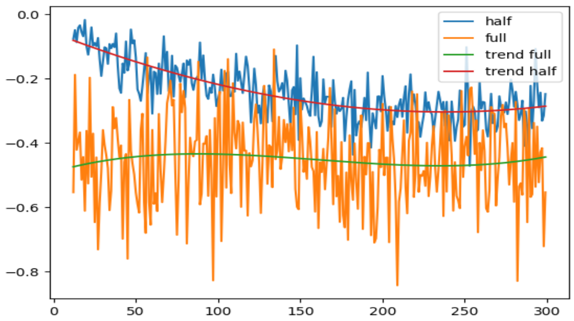

Figure 9: Comparison of Semantic shift between iPhone and X v.s. dimension of context vectors for representation proposed in Chatterjee and Sahoo (2013) and our representation for m=5, small data set

Cohen, T., Schvaneveldt, R., Widdows, D., 2010. Reflective random indexing and indirect inference: A scalable method for discovery of implicit connections. Journal of biomedical informatics 43, 240–256.

Joshi, A., Halseth, J.T., Kanerva, P., 2014. Language recognition using random indexing. ArXiv abs/1412.7026.

Jurgens, D., Stevens, K., 2009. Event detection in blogs using temporal random indexing, in: Proceedings of the Workshop on Events in Emerging Text Types, pp. 9–16.

QasemiZadeh, B., 2015. Random indexing revisited, in: International Conference on Applications of Natural Language to Information Systems, Springer. pp. 437–442.

QasemiZadeh, B., Handschuh, S., 2015. Random indexing explained with high probability, in: Král, P., Matoušek, V. (Eds.), Text, Speech, and Dialogue, Springer International Publishing, Cham. pp. 414–423.

Sahlgren, M., 2005. An introduction to random indexing.

Sandin, F., Emruli, B., Sahlgren, M., 2016. Random indexing of multidimensional data. Knowledge and Information Systems 52, 267–290.

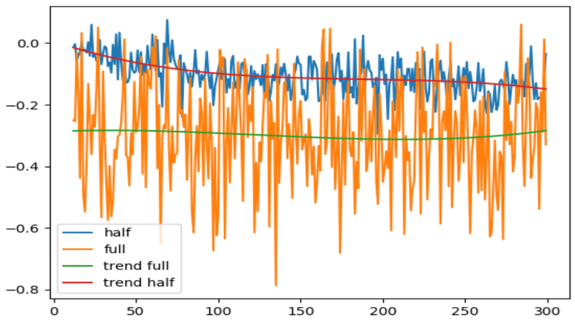

Figure 10: Comparison of Semantic shift between iPhone and X v.s. dimension of context vectors for representation proposed in Chatterjee and Sahoo (2013) and our representation for m=5, large data set

Appendix A.

The probability of orthogonality can be used to decide upon the dimension of the context vectors for a use case. We chose the minimum possible dimension which maximises the probability of orthogonality or such that it is above a threshold while being able to accommodate the word corpus. The following cut-off values of n are obtained from the code for the base and ri case, the size of RI space is also shown for the value of n (not mentioned in the paper), N = size of word corpus.The following observations are made to motivate the randomization of entries in { 1,-1}. (1)The cut-off value for a given probability is lesser for a given probability for ri case. (2) The corresponding size of the RI space is much larger for RI case. This gives the RI case an edge for decreasing the dimension of context vectors.These are shown in Tables-4, 5

Table 4: Cut-off values of n for base case

	$p > 90\%$	$p > 95\%$	$p > 97.5\%$
R=1	2	2	2
R=2	39	79	159
R=3	88	178	358
R=4	156	316	636
R=5	242	492	992
R=6	348	708	1000+

Table 5: Cut-off values of n for RI case along with word capacity

	$p > 90\%$	$p > 95\%$	$p > 97.5\%$
R=2	40 N= 3120	80 N= 12640	160 N=50880
R=3	87 N= 847960	177 N= 7268800	357 N= 60156880
R=4	153 N= 351165600	314 N= 6357666016	634 N= 106696002016
R=5	238 N= 195204469824	488 N= 7230043079424	988 N= 248514122298624
R=6	341 N= 133710757852672	701 N= 10323765985980160	1000+

Appendix B.

We compare the novel random indexing approach to that proposed in Chatterjee and Sahoo (2013). For this purpose, we plot the change in the angle between *X* and *iPhone* pre and post the launch of *iPhone X* to observe semantic shift. These are shown in Fig. 6, 10. We observe that the representation we chose to randomly index the context vector performs better at portraying the semantic shift as the amount of deviation is larger in this case but our representation is rather noisy accounting for the fact the +1's and -1's are distributed randomly instead of halves, so the performance is less stable, but the mean performance turns out to be better.To overcome the problem of reliability due to noisy performance, we may use majority voting of a cluster of context vectors, or the average of β iterations run over the same data set. The iterations will not take much time, as our algorithm runs in log linear time in the word corpus and hence will only be $O(\beta|W|\log(|W|))$ which is again log linear time in word corpus.

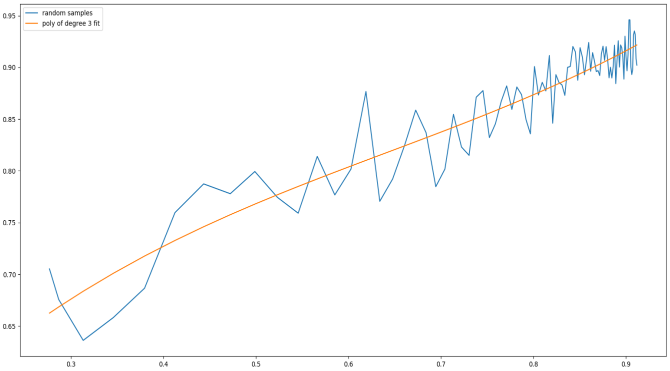

Figure B.11: Accuracy v.s Probability of orthogonality, cutoff n=300 for small data set with 600 unique words

Appendix C.

Here we present the effect of probability of orthogonality on the accuracy of the representations to convey meaningful semantic information about the relationship between different words in a given dataset. To examine the accuracy w.r.t. base case(BoW representations), probability of orthogonality is varied by varying "n" for a given "r". The results are randomized for a given probability, so for each probability point 20 samples are taken and averaged out, this is done for all probabilities. The metric used is : 1-$|\theta_i - \theta_0|$, where θ_i : the angle between initial and final semantic vectors of "iPhone" in RI case, θ_0: the angle between initial and final semantic vectors of "iPhone" in base case. This is shown in Fig. B.11, C.12 where a cubic trendline has also been plotted. We see that with increasing the probability of orthogonality the accuracy of the representations increase, however stochastically as the representations are random.

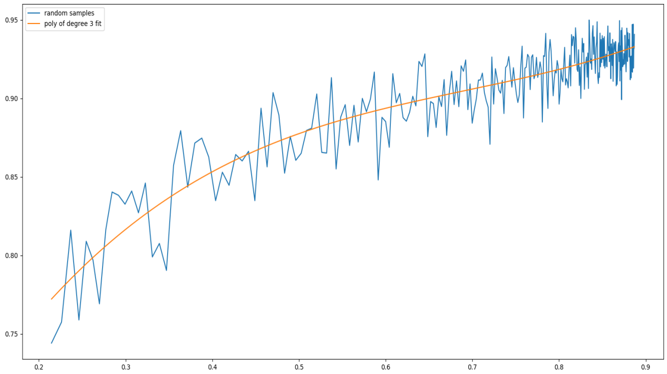

Figure C.12: Accuracy v.s Probability of orthogonality for cutoff n=300 for large data set with 2000 unique words