

—NODALIDA 2011—

Random Indexing Re-Hashed

Erik Velldal

Department of Informatics
University of Oslo, Norway
erikve@ifi.uio.no

May 12, 2011



Outline

1. Review **random indexing** for dimensionality reduction.
2. Review the notion of **universal families of hash functions**.
3. Show how $1 + 2 =$ **hashed random indexing**.



Outline

1. Review **random indexing** for dimensionality reduction.
2. Review the notion of **universal families of hash functions**.
3. Show how $1 + 2 =$ **hashed random indexing**.
4. Caveats.
5. Pilot experiments.
6. Summing up.



Random Indexing: Some History

- ▶ Initially intended as a **compact** way of modeling the semantic similarity of words in **word-by-document vector spaces** by Kanerva et al. (2000).
- ▶ Much work on RI-based **semantic spaces** has later followed (e.g. Karlgren & Sahlgren, 2001; Sahlgren, 2005).
- ▶ Many previous NODALIDA papers on RI;
 - ▶ Sahlgren and Swanberg (2001), Gambäck et al. (2003), Sahlgren (2003), Holmlund et al. (2005), Kann and Rosell (2005), Hassel and Sjöbergh (2007),...



Random Indexing: Some History

- ▶ Initially intended as a **compact** way of modeling the semantic similarity of words in **word-by-document vector spaces** by Kanerva et al. (2000).
- ▶ Much work on RI-based **semantic spaces** has later followed (e.g. Karlgren & Sahlgren, 2001; Sahlgren, 2005).
- ▶ Many previous NODALIDA papers on RI;
 - ▶ Sahlgren and Swanberg (2001), Gambäck et al. (2003), Sahlgren (2003), Holmlund et al. (2005), Kann and Rosell (2005), Hassel and Sjöbergh (2007),...
- ▶ Velldal (2010) applied RI for **SVM-based uncertainty classification**.



Random Indexing: Some History

- ▶ Initially intended as a **compact** way of modeling the semantic similarity of words in **word-by-document vector spaces** by Kanerva et al. (2000).
- ▶ Much work on RI-based **semantic spaces** has later followed (e.g. Karlgren & Sahlgren, 2001; Sahlgren, 2005).
- ▶ Many previous NODALIDA papers on RI;
 - ▶ Sahlgren and Swanberg (2001), Gambäck et al. (2003), Sahlgren (2003), Holmlund et al. (2005), Kann and Rosell (2005), Hassel and Sjöbergh (2007),...
- ▶ Velldal (2010) applied RI for **SVM-based uncertainty classification**.
- ▶ **Note:** While *not* here assuming any particular type of data or application, we will assume a **vector space model** for representation:
 - ▶ Given n examples and d features, the feature vectors can be thought of as rows in a matrix $F \in \mathfrak{R}^{n \times d}$.



Random Indexing

Goal

- ▶ Instead of using the original $n \times d$ feature matrix F , we will construct an $n \times k$ matrix G , where $k \ll d$.

Two Simple Steps

- ▶ As a new feature is instantiated, it is assigned a randomly generated **index vector**: A vector with a fixed dimensionality k , consisting of a small number of -1 s and $+1$ s, with the remaining elements set to 0.
- ▶ The vector representing a given training example (a row in G) is given by simply summing the random index vectors of its features.

Parameters

- ▶ The number of non-zeros (ϵ) and the dimensionality (k).



Constructing Feature Vectors: the Standard Approach

Features:

f_1 f_2 f_3 f_4 f_5 ...

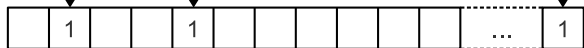
Feature activations:

+1

+1

+1

Feature vector $f(x)$:



Dimensions:

1 2 3 4 5 ...

d



Constructing Feature Vectors: the Standard Approach

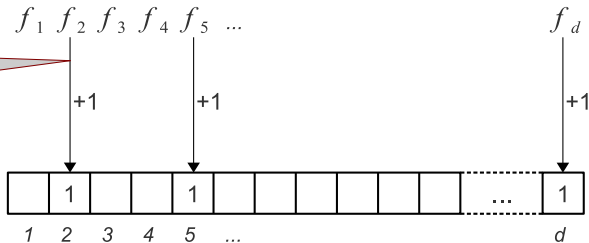
Features:

Each feature f_i maps to one dimension i

Feature activations:

Feature vector $f(x)$:

Dimensions:



Constructing Feature Vectors: the Standard Approach

Features:

Each feature f_i maps to *one* dimension i

f_1 f_2 f_3 f_4 f_5 ...

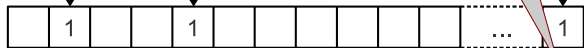
Feature activations:

+1

+1

+1

Feature vector $f(x)$:



Dimensions:

1 2 3 4 5 ...

d



Constructing Feature Vectors: the Standard Approach

Features:

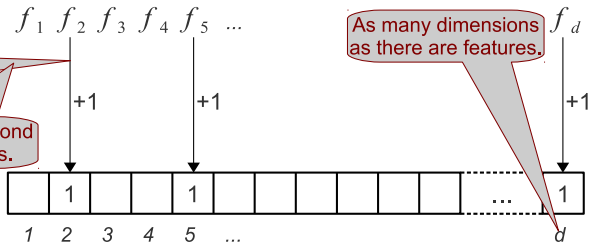
Each feature f_i maps to *one dimension i*

Feature activations:

The mappings correspond to *orthogonal vectors*.

Feature vector $f(x)$:

Dimensions:



Constructing Feature Vectors: the Standard Approach

Features:

Each feature f_i maps to *one* dimension i

f_1 f_2 f_3 f_4 f_5 ...

Feature activations:

The mappings correspond to *orthogonal* vectors.

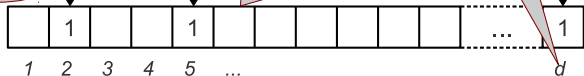
+1

The feature vector of a given example $f(x)$ is simply the sum of its active features.

As many dimensions as there are features.

+1

Feature vector $f(x)$:



Dimensions:



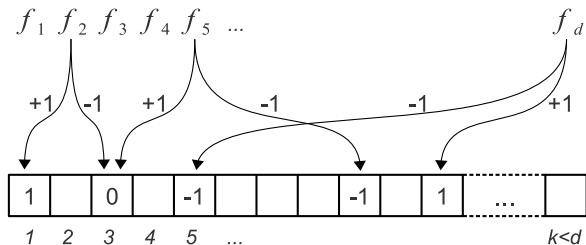
Constructing Feature Vectors: the RI Approach

Features:

Feature activations:

Feature vector $f(x)$:

Dimensions:



Constructing Feature Vectors: the RI Approach

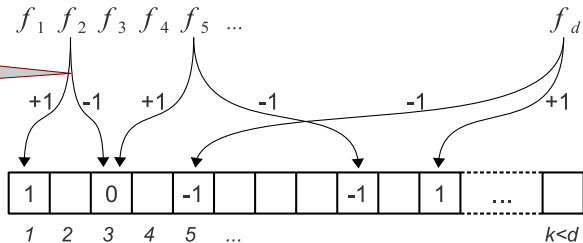
Features:

Each feature f_i is randomly mapped to several dimensions, valued -1 or +1.

Feature activations:

Feature vector $f(x)$:

Dimensions:



Constructing Feature Vectors: the RI Approach

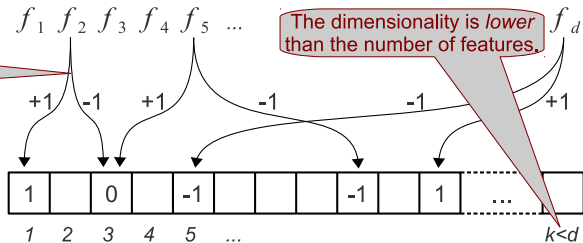
Features:

Each feature f_i is randomly mapped to several dimensions, valued -1 or +1.

Feature activations:

Feature vector $f(x)$:

Dimensions:



Constructing Feature Vectors: the RI Approach

Features:

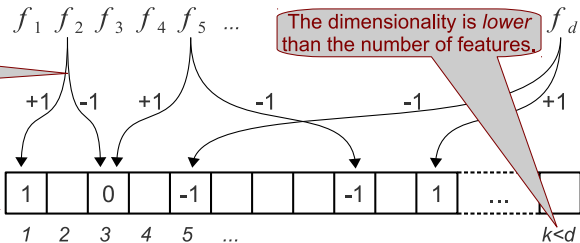
Each feature f_i is randomly mapped to several dimensions, valued -1 or +1.

Feature activations:

Feat

The mappings correspond to nearly orthogonal vectors (= the index vectors).

Dimensions:



Constructing Feature Vectors: the RI Approach

Features:

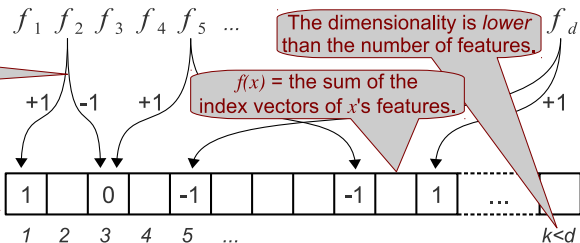
Each feature f_i is randomly mapped to several dimensions, valued -1 or +1.

Feature activations:

The mappings correspond to nearly orthogonal vectors (= the index vectors).

Feature

Dimensions:



RI—an example of Random Projections

- ▶ For $F \in \mathfrak{R}^{n \times d}$ and a random matrix $R \in \mathfrak{R}^{d \times k}$, where $k \ll d$:

$$FR = G \in \mathfrak{R}^{n \times k}$$

- ▶ The pairwise distances in F can be preserved in G with high probability (**the Johnson-Lindenstrauss lemma**).
- ▶ The rand. index of the i th feature corresponds to the i th row of R .



RI—an example of Random Projections

- ▶ For $F \in \mathfrak{R}^{n \times d}$ and a random matrix $R \in \mathfrak{R}^{d \times k}$, where $k \ll d$:

$$FR = G \in \mathfrak{R}^{n \times k}$$

- ▶ The pairwise distances in F can be preserved in G with high probability (**the Johnson-Lindenstrauss lemma**).
- ▶ The rand. index of the i th feature corresponds to the i th row of R .

A particular advantage of RI

- ▶ Constructs G by *incrementally* accumulating the index vectors.
 - ▶ Means that F does not need to be explicitly computed.
 - ▶ Constructs G directly (dimension reduction only implicit).
 - ▶ Can easily add more data without recomputing R and G .
 - ▶ Suitable for parallelization and stream processing.



Rethinking the Random Index Representation

- ▶ Storage is fairly cheap: For each index vector we only need to keep track of the signs and the positions of the non-zeros.
- ▶ Still, for hundreds of thousands or millions of features, it adds up. . .



Rethinking the Random Index Representation

- ▶ Storage is fairly cheap: For each index vector we only need to keep track of the signs and the positions of the non-zeros.
- ▶ Still, for hundreds of thousands or millions of features, it adds up. . .
- ▶ Taking a step back, the index vectors are reminiscent of probabilistic data structures like **Bloom Filters**. . .
 - ▶ Hashed-based data structure for compactly representing set membership.



Rethinking the Random Index Representation

- ▶ Storage is fairly cheap: For each index vector we only need to keep track of the signs and the positions of the non-zeros.
- ▶ Still, for hundreds of thousands or millions of features, it adds up. . .
- ▶ Taking a step back, the index vectors are reminiscent of probabilistic data structures like **Bloom Filters**. . .
 - ▶ Hashed-based data structure for compactly representing set membership.
- ▶ Idea: **We can save resources by having a set of hash functions compute and represent the index vectors.**
 - ▶ Eliminates the need for storing R .



Hashing

- ▶ For some set of hash **keys** $U = \{x_1, \dots, x_k\}$, a **hash function** h maps each x_i into some smaller set of hash **codes** $I = \{i_1, \dots, i_l\}$.
 - ▶ $h : U \rightarrow I$ with $|U| \geq |I|$.
- ▶ We can use hashing to implement the **compression** of RI;
 - ▶ The **keys** U are dimensions in the **original** space.
 - ▶ The **codes** I are dimensions in the **lower-dimensional** space.



Hashing

- ▶ For some set of hash **keys** $U = \{x_1, \dots, x_k\}$, a **hash function** h maps each x_i into some smaller set of hash **codes** $I = \{i_1, \dots, i_l\}$.
 - ▶ $h : U \rightarrow I$ with $|U| \geq |I|$.
- ▶ We can use hashing to implement the **compression** of RI;
 - ▶ The **keys** U are dimensions in the **original** space.
 - ▶ The **codes** I are dimensions in the **lower-dimensional** space.
- ▶ **Collisions**; multiple keys may be mapped to the same hash code.
 - ▶ Need to distribute codes as evenly as possible to reduce the noise.
- ▶ RI uses one-to-many mappings, so **we need *multiple* hash functions**.



Universal Families of Hash Functions

- ▶ Introduced by Carter and Wegman (1979).
- ▶ A method for **randomly generating hash functions** $h_i : U \rightarrow I$ from a family of functions H that guarantees that the **probability of a collision** for any two distinct keys is **bounded** by $1/|I|$.
- ▶ On demand, we can randomly select deterministic functions from H that maps the data to indices/codes as if at random.
- ▶ There exists several ways of implementing such universal classes. . .



Multiplicative Universal Hashing (Dietzfelbinger et al., 1997)

- ▶ A particularly simple class of mappings from k -bit keys to l -bit indices.
 - ▶ Let $U = \{0, \dots, 2^k - 1\}$ and $I = \{0, \dots, 2^l - 1\}$.
 - ▶ Let $A = \{a \mid 0 < a < 2^k \text{ and } a \text{ is odd}\}$.
 - ▶ Now $H_{k,l} = \{h_a \mid a \in A\}$ defines a 2-universal family where

$$h_a(x) = (ax \bmod 2^k) \operatorname{div} 2^{k-l} \quad \text{for } 0 \leq x < 2^k$$

- ▶ For two distinct keys x and y in U , h_a obeys

$$\operatorname{Prob}(h_a(x) = h_a(y)) \leq \frac{1}{2^{m-1}}$$

- ▶ By randomly picking a number $a \in A$ we generate a new hash function h_a from the set of 2^{k-1} distinct hash functions in $H_{k,l}$.
- ▶ Efficient bit-level implementation of modulo and integer division.



Hashed Random Indexing

- ▶ Any set of random index vectors with ϵ non-zeros in each can now be **implicitly represented** by a set of ϵ functions $\{h_{a^1}, \dots, h_{a^\epsilon}\} \subset H_{k,l}$.
- ▶ Half of the functions indicate -1 s and the other $+1$ s.



Hashed Random Indexing

- ▶ Any set of random index vectors with ϵ non-zeros in each can now be **implicitly represented** by a set of ϵ functions $\{h_{a^1}, \dots, h_{a^\epsilon}\} \subset H_{k,l}$.
- ▶ Half of the functions indicate -1 s and the other $+1$ s.
- ▶ Eliminates the $R \in \mathbb{R}^{d \times k}$ random matrix:
 - ▶ Store ϵ integers instead of the $d\epsilon$ signed positions minimally required otherwise.
- ▶ Can compute $FR = G$ without explicitly representing neither F or R .



Hashed Random Indexing

- ▶ Any set of random index vectors with ϵ non-zeros in each can now be **implicitly represented** by a set of ϵ functions $\{h_{a^1}, \dots, h_{a^\epsilon}\} \subset H_{k,l}$.
- ▶ Half of the functions indicate -1 s and the other $+1$ s.
- ▶ **Eliminates the $R \in \mathbb{R}^{d \times k}$ random matrix:**
 - ▶ Store ϵ integers instead of the $d\epsilon$ signed positions minimally required otherwise.
- ▶ Can compute $FR = G$ without explicitly representing neither F or R .
- ▶ Better support for parallelization:
 - ▶ The only knowledge that needs to be shared is the seed numbers.



Caveats

- ▶ Random projection methods (such as RI) are often applied for reducing memory load and computational cost. . .
- ▶ However, if your original space F is very *sparse*, the dimensionality reduction might give you the opposite effect.



Caveats

- ▶ Random projection methods (such as RI) are often applied for reducing memory load and computational cost. . .
- ▶ However, if your original space F is very *sparse*, the dimensionality reduction might give you the opposite effect.
- ▶ Why?
 - ▶ Because the reduced space G will then be much more *dense* than F ,
 - ▶ and the cost of storage and standard vector operations depend not on dimensionality alone, but on the number of **non-zero elements**.
 - ▶ Zero-valued elements can be ignored.



Pilot Experiments with Applying HRI

- ▶ Joint work with Lilja Øvrelid (University of Oslo) and Fredrik Jørgensen (Meltwater News).
- ▶ Two SVM-based classification tasks with large feature spaces:



Pilot Experiments with Applying HRI

- ▶ Joint work with Lilja Øvrelid (University of Oslo) and Fredrik Jørgensen (Meltwater News).
- ▶ Two SVM-based classification tasks with large feature spaces:
 - ▶ Stacked dependency parsing (Maltparser) on the Tiger treebank:
 - ▶ Features: 500,000 \rightarrow 16,384 ($\epsilon = 4$)
 - ▶ UAS: 90.15 \rightarrow 90.00
 - ▶ LAS: 87.83 \rightarrow 87.65
 - ▶ Uncertainty detection on the CoNLL-2010 shared task data:
 - ▶ Feature reduction: 670,000 \rightarrow 8,192 ($\epsilon = 4$)
 - ▶ Sentence-level F_1 : 86.78 \rightarrow 86.91



Pilot Experiments with Applying HRI

- ▶ Joint work with Lilja Øvrelid (University of Oslo) and Fredrik Jørgensen (Meltwater News).
- ▶ Two SVM-based classification tasks with large feature spaces:
 - ▶ Stacked dependency parsing (Maltparser) on the Tiger treebank:
 - ▶ Features: 500,000 \rightarrow 16,384 ($\epsilon = 4$)
 - ▶ UAS: 90.15 \rightarrow 90.00
 - ▶ LAS: 87.83 \rightarrow 87.65
 - ▶ Uncertainty detection on the CoNLL-2010 shared task data:
 - ▶ Feature reduction: 670,000 \rightarrow 8,192 ($\epsilon = 4$)
 - ▶ Sentence-level F_1 : 86.78 \rightarrow 86.91
- ▶ Feature space reduced by up to two orders of magnitude without statistically significant differences in classifier accuracy!



Summing up:

- ▶ Random Indexing:
 - ▶ Incremental random projection method.
 - ▶ Reduced and bounded dimensionality.
 - ▶ No need to explicitly represent the full input matrix.



Summing up:

- ▶ Random Indexing:
 - ▶ Incremental random projection method.
 - ▶ Reduced and bounded dimensionality.
 - ▶ No need to explicitly represent the full input matrix.
- ▶ Hashed Random Indexing:
 - ▶ Efficient reformulation of RI.
 - ▶ No need to explicitly represent the random vectors.
 - ▶ Rely on **universal hashing** instead.



Summing up:

- ▶ Random Indexing:
 - ▶ Incremental random projection method.
 - ▶ Reduced and bounded dimensionality.
 - ▶ No need to explicitly represent the full input matrix.
- ▶ Hashed Random Indexing:
 - ▶ Efficient reformulation of RI.
 - ▶ No need to explicitly represent the random vectors.
 - ▶ Rely on **universal hashing** instead.
- ▶ Hashing—an emerging trend in NLP!
 - ▶ Several recent studies the use of hashing for scaling up models.
 - ▶ Locality sensitive hashing, sketching, generalized bloom filters, hash-kernels, the hashing-trick, random feature mixing. . .
 - ▶ The relation to HRI further discussed in Velldal (2011).



- Carter, L., & Wegman, M. N. (1979). Universal classes of hash functions. *Journal of Computer and System Sciences*, 18(2), 143–154.
- Dietzfelbinger, M., Hagerup, T., Katajainen, J., & Penttonen, M. (1997). A reliable randomized algorithm for the closest-pair problem. *Journal of Algorithms*, 25(1), 19–51.
- Gambäck, B., Cheadle, M., Hansen, P., Olsson, F., & Sahlgren, M. (2003). A spoken swedish e-mail interface. In *Proceedings of the 14th Nordic Conference on Computational Linguistics*. Reykjavík, Iceland.
- Hassel, M., & Sjöbergh, J. (2007). Widening the holsum search scope. In *Proceedings of the 16th Nordic Conference on Computational Linguistics*. Tartu, Estonia.
- Holmlund, J., Sahlgren, M., & Karlgren, J. (2005). Creating bilingual lexica using reference wordlists for alignment of monolingual semantic vector spaces. In *Proceedings of the 15th Nordic Conference on Computational Linguistics*. Joensuu, Finland.
- Kanerva, P., Kristoferson, J., & Holst, A. (2000). Random indexing of text samples for latent semantic analysis. In *Proceedings of the 22nd annual conference of the cognitive science society* (p. 1036). Pennsylvania.
- Kann, V., & Rosell, M. (2005). Free construction of a free swedish dictionary of synonyms. In *Proceedings of the 15th Nordic Conference on Computational Linguistics*. Joensuu, Finland.
- Karlgren, J., & Sahlgren, M. (2001). From words to understanding. In Y. Uesaka, P. Kanerva, & H. Asoh (Eds.), *Foundations of real-world intelligence* (pp. 294–308). Stanford: CSLI Publications.
- Sahlgren, M. (2003). Content-based adaptivity in multilingual dialogue systems. In *Proceedings of the 14th Nordic Conference on Computational Linguistics*. Reykjavík, Iceland.
- Sahlgren, M. (2005). An introduction to random indexing. In *Proceedings of the methods and applications of semantic indexing workshop at the 7th international conference on terminology and knowledge engineering (TKE)*. Copenhagen, Denmark.
- Sahlgren, M., & Swanberg, D. (2001). Using linguistic information to improve the performance of vector-based semantic analysis. In *Proceedings of the 13th Nordic Conference on Computational Linguistics*. Uppsala, Sweden.
- Velldal, E. (2010). Detecting uncertainty in biomedical literature: A simple disambiguation approach using sparse random indexing. In *Proceedings of the fourth international symposium on semantic mining in biomedicine (SMBM)*. Cambridgeshire, UK.
- Velldal, E. (2011). Random indexing re-hashed. In *Proceedings of the the 18th nordic conference of computational linguistics*. Riga, Latvia.

