Inherently Interpretable Sparse Word Embeddings through Sparse Coding

Adly Templeton
Williams College
adlytempleton@gmail.com

Abstract

Word embeddings are a powerful natural language processing technique, but they are extremely difficult to interpret. In order to create more interpretable word embeddings, we transform pretrained dense word embeddings into sparse embeddings. These new embeddings are inherently interpretable: each of their dimensions are created from and represent a natural language word or specific syntactic concept. We construct these embeddings through sparse coding, where each vector in the basis set is itself a word embedding. We show that models trained using these sparse embeddings can achieve good performance and are extremely interpretable.

1 Introduction

Word embeddings are a powerful tool , but it is very difficult to interpret a dimension of a typical word embedding space. This obscures interpretation of any models built on top of word embeddings.

To enable interpretable models, we seek vectors where each dimension is given an inherent 'interpretation'. To do this, we will represent word embeddings as the sparse linear combination of a basis set of other word embeddings.

2 Previous Work

Previous work has created more interpretable word embeddings in different ways.

Park et al. [Park et al., 2017] find a more interpretable rotation of word embeddings using techniques typically associated with factor analysis. Other work [Dufter and Schütze, 2019; Rothe and Schütze, 2016] rotates dense vectors using different methods.

Koc et al. [Koç et al., 2018] tie concepts to dimensions in a more direct way. They select a concept for each dense dimension and identify words that are associated with these concepts. Then, a penalty term pushes the coefficients for these words towards the appropriate values in those dimensions.

Other work has focused on interpretability through sparsity. Subramian et al. [Subramanian et al., 2018] created more interpretable embeddings by passing pretrained dense embeddings through a sparse autoencoder. Panigrahi et al. proposed Word2Sense, a generative approach that models each dimension as a 'sense' and word embeddings as a sparse probability distribution over the senses [Panigrahi et al., 2019].

The representation of vectors as the sparse linear combination of a basis, called 'sparse coding', is a well-studied optimization problem [Coates and Ng, 2011; Hoyer, 2002; Lee et al., 2007]. Previous work [Coates and Ng, 2011] has also shown that basis vectors can be selected from the set that is being encoded without significantly decreasing efficiency.

Faruqui et al.[Faruqui et al., 2015] used non-negative sparse coding to recode dense word embeddings into more interpretable sparse vectors while learning a basis.

Zhang et al. [Zhang et al., 2019] also used nonnegative sparse coding to learn a set of *word factors* to recode word2vec embeddings. The basis vectors created in this way are highly redundant, so they then use spectral clustering to remove near-duplicate factors. Then, they are able to manually infer reasonable post hoc interpretations for most of the factors.

Concurrently with our work, Matthew et al. create an inherently interpretable subspace from pairs of antonyms. They then project dense word embeddings into that subspace, producing lower-dimensional dense vectors [Mathew et al., 2020].

3 Model

Our work uses *sparse coding* to transform a set of word embeddings from a dense and uninterpretable space into a sparse, interpretable, space. Let v_D represent a dense word embedding, and let $\mathcal{B}$ represent a matrix with 'basis' vectors along the columns. We achieve sparse coding using regularized regression, inducing sparsity using the L_1 norm. Formally, this corresponds to finding the sparse vector v_S that minimizes the following objective function

$$\arg\min_{v_S} \; ||v_D - v_S\mathcal{B}||_2^2 + \alpha||v_s||_1 \qquad (1)$$

α is a hyperparameter that controls the level of sparsity. Our key innovation is that we draw the columns of the basis from the original set of dense word embeddings. This strategy provides a natural label for each dimension in the sparse space.

3.1 Syntactic Basis

We can roughly divide the 'meaning' carried by a word embedding into *syntactic* and *semantic* properties. Here we use 'syntactic properties' to mean properties that describe how that word fits into the grammar of the language, such as part-of-speech, tense, or number. By contrast, we use 'semantic properties' to mean all other aspects of the meaning of a word. For instance, we expect the embedding for the word ‘swimming’ to include a syntactic component denoting that this word is a present-tense participle and a semantic component that represents the meaning ‘to swim’. Of course, this deconstruction is imperfect. Nevertheless, this approach provides a useful insight towards decomposing the meaning of a word embedding.

Intuition suggests that word-labeled dimensions poorly capture the syntactic properties of word embeddings. Preliminary experiments confirmed that, without special consideration, syntactic properties would be captured by an uninterpretable combination of unintuitive dimensions. To address this, we construct a small number of *syntactic basis vectors* and add them to the basis set. For instance, we construct a ‘POS-NOUN’ vector by taking the mean of all word embeddings corresponding to nouns. A full description of the syntactic basis vectors is in the appendix.

Next, we make the syntactic basis vectors orthogonal using the Gram-Schmidt process. Finally, we subtract the projection along the syntactic basis vectors from all other (‘semantic’) basis vectors we use, and renormalize them. This procedure separates the syntactic meaning from our semantic basis vectors, ensuring that semantic bases are not also coding for syntactic concepts.

When we are encoding a dense vector, instead of encoding the syntactic dimensions using sparse coding, we set each syntactic coefficient to the projection along that vector (which is equal to the similarity between the original vector and the syntactic basis vector). This residual is then transformed using Equation 1.

3.2 Basis Selection

We cannot practically use all words as our basis set, so we have to select a subset. First, we start with the 30,000 most frequent words. We filter out any words that are capitalized or that are not in a standard English dictionary. Next, we filter out any words that are not nouns, verbs, or adjectives. This process removes many basis vectors that may be hard to interpret. This gives us approximately 11,000 remaining potential basis words. From these, we will select 3,000 words to use in the final basis.

We use an iterative algorithm that takes, at each step, the potential basis vector with the highest mean cosine similarity to all other vectors. To encourage diversity, this mean is weighted by the lowest cosine dissimilarity that each vector has with an already-selected basis vector.

4 Results

4.1 Comparison with previous work

To compare our work against other sparse coding approaches, we will often reference the vectors created

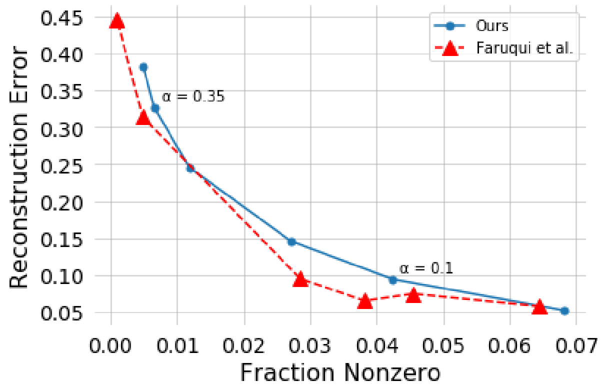

Figure 1: The tradeoff curve between sparsity and reconstruction error. The solid line shows the tradeoff curve achieved by our models. The dashed line shows the tradeoff curve achieved by Faruqui et al. using sparse coding without inherently interpretable dimensions (Section 4.1)

by Faruqui et al.[Faruqui et al., 2015]. That work generates more interpretable vectors using sparse coding, but without inherently interpretable dimensions.

4.2 Reconstruction Error and Sparsity

Note that, because of the penalty term in Equation 1, $V_S\mathcal{B}$ (the *reconstructed vectors*) are not exactly equal to the original dense vectors V_D. Therefore, we expect a tradeoff between sparsity and this difference (which we call *reconstruction error).*

This tradeoff curve is displayed in Figure 1. Measuring this tradeoff gives a useful task-agnostic measure of performance. Comparing to the sparse coding of Faruqui et al., we can see that, despite the additional constraints of an inherently interpretable system, we suffer only a minor increase in reconstruction error compared to traditional sparse coding.

For the remainder of this paper, unless otherwise mentioned, we will consider the vectors made with $\alpha = 0.35$.

4.3 Analogy Task

The Mikolov et al. [Mikolov et al., 2013] paper first detailing word2vec measured the performance of their system by showing that it could successfully complete *word analogy tasks*. In the word2vec vector space, famously, the vector for 'king' plus the vector for 'woman' minus the vector for 'man' is close to the vector for 'queen'. Analogy tasks quantitatively tests these properties.

The task consists of analogies of the form *A is to A' as B is to B'*. The vector space is evaluated on its ability to correctly determine the value of B'.

The performance of our vector space at this task is displayed in Table 1. In general, our model performs poorly on this task. This degradation comes from two sources: First, the drop from the original vectors to the reconstructed vectors due to reconstruction error. Second, an additional degradation is caused by the transformation from dense vectors to sparse vectors. The method used for the analogy task is based on cosine similarity, which is noisy with sparse vectors.

	Nonzero	Total	Syn.	Sem.
FastText (centered)	300	0.88	0.85	0.94
Faruqui $\lambda = 0.75$	136	0.65	0.60	0.73
Ours $\alpha = 0.1$	127	0.50	0.46	0.59
Ours Recons. $\alpha = 0.1$	300	0.83	0.80	0.88
Ours $\alpha = 0.35$	20	0.20	0.22	0.15
Ours Recons. $\alpha = 0.35$	300	0.33	0.38	0.25

Table 1: Accuracy on the word2vec analogy evaluation set for various vector spaces. The first column shows the average number of nonzero entries in each sparse vector. Accuracy is also broken down by semantic and syntactic categories. All results are on a 50% held-out test set. For the comparison with the original dense vectors (FastText [Bojanowski et al., 2016]), we subtract the mean of all vectors, to match preprocessing.

4.4 Classification

Next, we demonstrate that our model can be used to build interpretable machine learning systems. To this end, we train classifiers using our word embeddings as an input. We demonstrate that these classifiers are not only effective but also interpretable.

We evaluate our vectors on two datasets, the IMDB sentiment analysis dataset [Maas et al., 2011] and the TREC question classification dataset [Li and Roth, 2002]. For both of these datasets, we use the same model. Both datasets involve reading in an English sentence (or sentences) and placing it into one of several categories. We use a bag-of-words approach that adds together the vector representations of each word in the sentence, ignoring capitalization. Then we train a logistic regression model on this fixed-length representation of each sentence. The logistic regression model is regularized using a L^2 penalty, which is found by cross-validation on the training set.

4.4.1 IMDB Sentiment Analysis Dataset

The IMDB movie review dataset consists of 50,000 passages taken from IMDB movie reviews, evenly split between positive and negative reviews. The task is to determine whether each passage expresses positive or negative sentiment about the movie it describes [Maas et al., 2011].

We train classifiers using various word embedding spaces as inputs. The results are presented in Table 2. Our vector spaces demonstrate improvement over the original dense vectors (FastText [Bojanowski et al., 2016]), as well as the traditional sparse coding approach of Faruqui et al. This result holds despite a slight decrease in performance caused by the reconstruction error (as demonstrated by the low performance with reconstructed vectors).

Unlike a classifier trained on the original vector space, we can interpret our classifier's coefficients. Here, we present the most significant coefficients ($\alpha = 0.1$):

$$\ln \frac{P(\text{positive})}{1 - P(\text{positive})} = -157 \cdot \text{dreadful}$$
$$- 153 \cdot \text{horrible} + 150 \cdot \text{fabulous} - 140 \cdot \text{dull}$$
$$- 132 \cdot \text{dreary} - 107 \cdot \text{worsen}$$
$$- 105 \cdot \text{ridiculous} + ...$$

Note that these are not coefficients on the frequencies of individual words. Instead, these are coefficients on vectors in the basis set. We can consider them to be coefficients on concepts, which are labeled by the displayed words. The coefficients make sense: positive concepts have positive coefficients, while negative concepts have negative coefficients. This pattern continues for much longer than displayed above, and we have omitted other terms for space reasons. The first term to not fall into this clear interpretation is the 24th-most significant: ... $+ 74 \cdot$ shall $+$ ..‘

The top five words in the dimension represented by ‘shall’ are the following: ‘henceforth’, ‘herein’, ‘hereafter’, ‘thereof’, ‘hereby’. This dimension appears to correspond to a formal and somewhat archaic tone, which is likely not found in a negative internet comment. This example demonstrates the potential for our interpretable vector space to reveal non-trivial details about how a classifier makes decisions.

	IMDB	TREC
FastText	85.35	84.2
Faruqui $\lambda = .75$	85.54	84.4
Ours $\alpha = 0.1$	**87.51**	**86.2**
Ours Recons. $\alpha = 0.1$	85.08	81.4
Ours $\alpha = .35$	86.46	84.0
Ours Recons. $\alpha = .35$	83.00	75.8

Table 2: Accuracy on the IMDB sentiment analysis dataset and the TREC question classification dataset. We use a logistic regression classifier, which uses as input a bag-of-words sum of various word embeddings.

4.4.2 TREC Question Classification Dataset

The TREC question classification dataset consists of 6,000 questions that are divided into 6 categories based on the expected answer: abbreviations, descriptions, entities, humans, locations, and numeric.

Accuracy for various vector spaces is presented in Table 2. Again, our model does better than the unmodified input vectors we start with, despite some loss from the reconstruction error. Both results suggest that our vector spaces are efficient in regression-based settings, even though the performance at the word-analogy task suffers a serious degradation.

Once again, we directly interpret the coefficients learned by logistic regression. For space, we display the most significant terms for the HUM category. Questions in this category expect the name of a human as the answer:

$$\ln \frac{P(\text{HUM})}{1-P(\text{HUM})} = -77 \cdot \text{wonder}$$
$$+ 66 \cdot \text{organizations} + 54 \cdot \text{companies} + 51 \cdot \text{poet}$$
$$+ 49 \cdot \text{songwriter} + 48 \cdot \text{identities} + 42 \cdot \text{fan}$$
$$- 42 \cdot \text{movie} - 39 \cdot \text{resulting} + 36 \cdot \text{university}$$
$$- 36 \cdot \text{diseases} + 36 \cdot \text{successive} + 35 \cdot \text{consist}$$
$$+ 35 \cdot \text{cabinets} + ...$$

Some of these coefficients, such as 'songwriter' or 'identities' are intuitive and reveal interesting behavior of the classifier. Others, such as 'wonder', are not. Manual inspection reveals that 'wonder' is used to represent words such as 'How' or 'why' but not 'who', though this behavior is likely noise.

4.4.3 Word Intrusion Task

To quantitatively measure interpretability, we use human experiments. In particular, we use the word intrusion task [Chang et al., 2009]. In this task, humans are presented with five words, four of which are associated highly with a particular dimension. Participants are asked to choose the word that does not belong.

We use the following procedure for generating questions. First, we filter candidate words, starting with the 20,000 most frequent words and filtering out words which are not lowercase, words which are not made up of only ASCII alphabetic characters, and words with only one letter. Then we randomly select a dimension. We pick the 4 highest words along that dimension, and one word randomly selected from the bottom 50% of words in that dimension, then randomize the order. We use our vectors both with and without providing the label of the dimension as a 'hint'. Each example is presented to three different Mechanical Turk annotators.

	Accuracy	**CI**
FastText	0.31 (253/826)	[0.27,0.34]
Faruqui	0.77 (639/829)	[0.74,0.80]
Ours	0.80 (661/827)	[0.77,0.83]
Ours + Hints	0.84 (690/ 822)	[0.81,0.86]

Table 3: Results on the word intrusion task. 95% normal confidence intervals are displayed.

The results of the word intrusion task are presented in Table 3. In particular, when hints are provided, we see a statistically significant improvement in accuracy between our vectors and the sparse coding baseline (p = .00055). In addition, using hints produces a statistically significant improvement (p = .040), validating our motivation for inherently interpretable dimensions.

5 Analysis

5.0.1 Frequently Used Semantic Basis Vectors

Next, we will manually analyze some semantic basis dimensions. Because we cannot practically investigate every one of the 2,989 semantic basis dimensions, we sample basis dimensions that are unusual or interesting in various ways.

For $\alpha = 0.35$, the five most frequently used semantic basis vectors are 'countries', 'namespace', 'page', 'imposes', and 'cruft'. 'Countries' is used to represent a lot of countries, while 'namespace' is used for internet-related terms and 'page' for terms related to Wikipedia metadata. 'Imposes' differs from the basis vectors we have examined so far: It is used to represent a lot of frequent verbs with somewhat-related meanings in a fairly ordinary way. For example, it is very heavily used in 'carries', 'puts', 'limits', and 'introduces'. 'Cruft' is used to represent both a lot of technical terms and a lot of fantasy terms, as well as various acronyms.

Overall, most of these vectors can be considered an artifact of the filter in our basis selection process, as other potential basis vectors that could be used to represent these types of words are filtered out.

5.0.2 Basis Vectors that were selected early

Recall that we use an iterative process to select basis vectors, suggesting that the 'best' basis vectors may be chosen first. The first vectors to be selected appear to be remarkably intuitively good concepts to use as a basis. The first ten selected basis vectors are 'beautiful', 'young', 'murdering', 'government', 'gorgeous', 'leaders', 'optimized', 'clergyman', 'antics', and 'charming'. Furthermore, all of these dimensions are used in extremely intuitive ways. For instance, 'beautiful' is used to represent 'attractive', 'scenic', and 'Marilyn' (the latter presumably refers to Marilyn Monroe).'young' is used to represent various age-related terms, and 'murdering' is used to represent various crimes.

6 Future Work

Our method still has some serious drawbacks. Sparse coding, by its nature, introduces a substantial amount of noise (reconstruction error). In addition to the reconstruction error, sparse coding can add a lot of noise because it can assign very different sparse vectors to similar dense vectors. We hope that future work can produce (sparse) embeddings that are interpretable by construction without some of the shortcomings of our work, either by making changes to the sparse coding method or by using a different method entirely.

References

Mathieu Blondel and Fabian Pedregosa. 2016. Lightning: large-scale linear classification, regression and ranking in Python.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*.

Rakesh Chalasani, Jose C Principe, and Naveen Ramakrishnan. 2013. A fast proximal method for convolutional sparse coding. In *The 2013 International Joint Conference on Neural Networks (IJCNN)*, pages 1–5. IEEE.

Jonathan Chang, Sean Gerrish, Chong Wang, Jordan L Boyd-Graber, and David M Blei. 2009. Reading tea leaves: How humans interpret topic models. In *Advances in Neural Information Processing Systems*, pages 288–296.

Adam Coates and Andrew Y Ng. 2011. The importance of encoding versus training with sparse coding and vector quantization. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 921–928.

Philipp Dufter and Hinrich Schütze. 2019. Analytical methods for interpretable ultradense word embeddings. *arXiv preprint arXiv:1904.08654*.

Manaal Faruqui, Yulia Tsvetkov, Dani Yogatama, Chris Dyer, and Noah Smith. 2015. Sparse overcomplete word vector representations. *arXiv preprint arXiv:1506.02004*.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

Patrik O Hoyer. 2002. Non-negative sparse coding. In *Proceedings of the 12th IEEE Workshop on Neural Networks for Signal Processing*, pages 557–565. IEEE.

Aykut Koç, Lutfi Kerem Senel, Ihsan Utlu, and Haldun M Ozaktas. 2018. Imparting interpretability to word embeddings while preserving semantic structure. *arXiv preprint arXiv:1807.07279*.

Thomas K Landauer and Susan T Dumais. 1997. A solution to plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological review*, 104(2):211.

Honglak Lee, Alexis Battle, Rajat Raina, and Andrew Y Ng. 2007. Efficient sparse coding algorithms. In *Advances in Neural Information Processing Systems*, pages 801–808.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *Proceedings of the 19th international conference on Computational linguistics-Volume 1*, pages 1–7. Association for Computational Linguistics.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Binny Mathew, Sandipan Sikdar, Florian Lemmerich, and Markus Strohmaier. 2020. The polar framework: Polar opposites enable interpretability of pre-trained word embeddings. *arXiv preprint arXiv:2001.09876*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119.

Ryosuke Okuta, Yuya Unno, Daisuke Nishino, Shohei Hido, and Crissman Loomis. 2017. CuPy: A NumPy-compatible library for nvidia gpu calculations. In *Proceedings of Workshop on Machine Learning Systems (LearningSys) in The Thirty-first Annual Conference on Neural Information Processing Systems (NIPS)*.

Travis E Oliphant. 2006. *A guide to NumPy*, volume 1. Trelgol Publishing USA.

Abhishek Panigrahi, Harsha Vardhan Simhadri, and Chiranjib Bhattacharyya. 2019. Word2sense: Sparse interpretable word embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5692–5705.

Sungjoon Park, JinYeong Bak, and Alice Oh. 2017. Rotated word vector representations and their interpretability. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 401–411.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in Python. *Journal of machine learning research*, 12(Oct):2825–2830.

Radim Rehurek and Petr Sojka. 2010. Software framework for topic modelling with large corpora. In *In Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. Citeseer.

Sascha Rothe and Hinrich Schütze. 2016. Word embedding calculus in meaningful ultradense subspaces. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 512–517.

Anant Subramanian, Danish Pruthi, Harsh Jhamtani, Taylor Berg-Kirkpatrick, and Eduard Hovy. 2018. Spine: Sparse interpretable neural embeddings. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Juexiao Zhang, Yubei Chen, Brian Cheung, and Bruno A Olshausen. 2019. Word embedding visualization via dictionary learning. *arXiv preprint arXiv:1910.03833*.

7 Appendix

A Syntacic Basis Descriptions

Our approach makes use of four types of syntactic basis vectors:

1. We use the first principal component of the embeddings of the 30,000 most frequent words. Previous work on word embedding has referred to this as the *common discourse vector*, or c_0, and has shown that this vector encodes words that appear commonly in all contexts, such as 'the'.

2. We use the mean of all vectors of capitalized words and use this as a syntactic basis vector to represent capitalization.

3. For a variety of parts-of-speech, we use the mean vectors for words with that part-of-speech (POS). Specifically, we encode a vector for each of the following: nouns, verbs, adjectives, adverbs, and numbers.

4. We create mean vector differences for the following syntactic concepts: the relationship between single and plural nouns, the relationship between present-tense verbs and their present participle form and the relationship between present-tense verbs and their past-tense forms. For each of these relationships, we manually collect approximately 50 example word pairs that fit that relationship. We manually filter for word pairs where either the syntactic relationship does not change the form of the word (ie. 'deer') or for word pairs where the syntactic change is likely to produce a more complicated change in meaning (ie. 'math' and 'maths'). We average the differences between pairs of each relationship type and use it as the vector for that relationship.

The choice and construction of these syntactic basis vectors is highly arbitrary, and different syntactic basis vectors could easily be used in different applications or in follow up work.

B Implementation

We use the FastText [Bojanowski et al., 2016] pretrained 300 dimensional English vectors (without subword information) trained on Wikipedia 2017, UMBC webbase corpus and statmt.org news dataset as the dense vectors which we input into our models. Unless otherwise mentioned, we only consider the 30,000 most frequent words, for computational reasons. We normalize all vectors to have mean 0 and unit length. After learning sparse vectors, we normalize the sparse vector so that the sparse vector corresponds to a dense vector of unit length.

In practice, the sparse penalty term will only make values very close to 0. We set any entry with an magnitude less than .001 to 0. We found this threshold by taking the lowest cutoff that does not introduce significant irregularities into the sparsity-error tradeoff curves in Section 4.2.

We solve the regularized optimization problems using the FISTA algorithm[Chalasani et al., 2013], as implemented in the Python Lightning package[Blondel and Pedregosa, 2016], using default hyperparameters. FISTA is an optimization algorithm which can efficiently solve sparse coding problems. We use the SpaCy library [Honnibal and Montani, 2017] to check for out of vocabulary words and perform part-of-speech tagging. We use the numpy[Oliphant, 2006], CuPy[Okuta et al., 2017], and Scikit learn [Pedregosa et al., 2011] libraries for various linear algebra implementations. We use the open-source Gensim library [Rehurek and Sojka, 2010] for various word embedding related tasks. For the word analogy task evaluation, we use the *3CosAdd* method, as implemented by Gensim.

B.1 Comparison to Faruqui et al.

To compare to the sparse coding approach of Faruqui et al. we use the following settings: we use their publicly available implementation with the following settings: We use the same input vectors without preprocessing, a dimensionality of 3000, L^2 regularization penalty $\tau = 10^{-5}$, as suggested in their paper, and various L^1 regularization penalties.

B.2 Randomly Selected Word Representations

We randomly select 25 words and display their complete sparse vector representations here:

$$\begin{aligned}
\text{carbon} &= 0.79 * \text{nitrogen} \\
&- 0.38 * \text{CAPITALIZATION} + 0.3 * \text{fossil} \\
&- 0.21 * \text{POS-NOUN} + 0.16 * \text{POS-ADJ} \\
&+ 0.14 * \text{C0} - 0.14 * \text{PAST-TENSE} \\
&+ 0.13 * \text{wood} + 0.11 * \text{global} \\
&+ 0.1 * \text{atoms} - 0.095 * \text{POS-ADV} \\
&+ 0.092 * \text{aluminum} - 0.078 * \text{PLURAL-NOUN} \\
&+ 0.073 * \text{greenhouse} - 0.072 * \text{POS-PROPN} \\
&- 0.048 * \text{POS-VERB} + 0.046 * \text{forestry} \\
&+ 0.03 * \text{PARTICIPLE} + 0.017 * \text{sink} \\
&+ 0.012 * \text{POS-NUM}
\end{aligned}$$

$\text{reefs} = 0.68 * \text{islands}$
$- 0.66 * \text{CAPITALIZATION} + 0.4 * \text{C0}$
$+ 0.35 * \text{PLURAL-NOUN} + 0.28 * \text{POS-VERB}$
$+ 0.25 * \text{rocks} + 0.19 * \text{dredging}$
$+ 0.18 * \text{oysters} + 0.12 * \text{POS-ADJ}$
$+ 0.096 * \text{POS-NUM} + 0.096 * \text{POS-ADV}$
$+ 0.089 * \text{POS-PROPN} + 0.086 * \text{tropical}$
$+ 0.075 * \text{underwater} + 0.068 * \text{dunes}$
$+ 0.063 * \text{seas} + 0.06 * \text{diver}$
$- 0.058 * \text{PAST-TENSE} + 0.042 * \text{sandstone}$
$- 0.025 * \text{demon} + 0.02 * \text{marine}$
$- 0.019 * \text{PARTICIPLE} - 0.014 * \text{POS-NOUN}$
$- 0.012 * \text{french} - 0.0041 * \text{witches}$

$\text{Coulson} = 0.85 * \text{hacking}$
$+ 0.72 * \text{C0} + 0.59 * \text{POS-PROPN}$
$+ 0.25 * \text{CAPITALIZATION} - 0.23 * \text{POS-ADJ}$
$- 0.19 * \text{POS-NOUN} + 0.17 * \text{butler}$
$- 0.17 * \text{southern} + 0.15 * \text{POS-VERB}$
$- 0.14 * \text{website} - 0.12 * \text{com}$
$- 0.12 * \text{roaring} + 0.1 * \text{solicitors}$
$- 0.094 * \text{POS-ADV} + 0.074 * \text{oats}$
$- 0.068 * \text{cathedral} - 0.064 * \text{PARTICIPLE}$
$+ 0.061 * \text{inquiry} - 0.06 * \text{dances}$
$- 0.056 * \text{fan} + 0.042 * \text{POS-NUM}$
$- 0.029 * \text{provinces} - 0.029 * \text{finals}$
$- 0.02 * \text{dance} - 0.017 * \text{waters}$
$- 0.013 * \text{tango} - 0.013 * \text{shame}$
$- 0.012 * \text{PAST-TENSE} - 0.005 * \text{PLURAL-NOUN}$

$\text{roundabout} = 0.72 * \text{bypass}$
$+ 0.4 * \text{roadway} - 0.28 * \text{CAPITALIZATION}$
$+ 0.22 * \text{plaza} - 0.16 * \text{PLURAL-NOUN}$
$+ 0.11 * \text{POS-ADJ} + 0.11 * \text{clumsy}$
$+ 0.1 * \text{airfield} + 0.088 * \text{POS-ADV}$
$- 0.08 * \text{biological} + 0.079 * \text{C0}$
$+ 0.051 * \text{POS-NOUN} + 0.043 * \text{PAST-TENSE}$
$+ 0.039 * \text{PARTICIPLE} + 0.028 * \text{caravan}$
$+ 0.025 * \text{ironic} + 0.021 * \text{POS-VERB}$
$- 0.021 * \text{POS-NUM} - 0.0038 * \text{POS-PROPN}$
$+ 0.003 * \text{nonsensical}$

$$\text{Hub} = 0.49 * \text{bustling} + 0.47 * \text{C0} + 0.4 * \text{portal} + 0.39 * \text{infrastructure} + 0.32 * \text{POS-NOUN} + 0.31 * \text{CAPITALIZATION} + 0.31 * \text{central} - 0.13 * \text{POS-PROPN} - 0.1 * \text{PLURAL-NOUN} + 0.069 * \text{outage} + 0.068 * \text{centre} + 0.061 * \text{POS-NUM} + 0.058 * \text{connectivity} + 0.058 * \text{PARTICIPLE} - 0.057 * \text{POS-ADJ} + 0.043 * \text{PAST-TENSE} - 0.043 * \text{POS-VERB} + 0.027 * \text{POS-ADV}$$

$$\text{Churchill} = 0.84 * \text{wartime} + 0.6 * \text{C0} + 0.41 * \text{CAPITALIZATION} + 0.4 * \text{quotation} + 0.38 * \text{POS-PROPN} + 0.36 * \text{statesman} - 0.21 * \text{PARTICIPLE} - 0.14 * \text{astronomer} - 0.14 * \text{POS-NOUN} - 0.11 * \text{POS-ADJ} - 0.1 * \text{PAST-TENSE} + 0.082 * \text{POS-VERB} + 0.078 * \text{POS-NUM} + 0.064 * \text{POS-ADV} + 0.045 * \text{advising} - 0.025 * \text{architectures} + 0.022 * \text{PLURAL-NOUN} + 0.017 * \text{pint} + 0.013 * \text{fascism}$$

$$\text{environmental} = 0.43 * \text{sustainability} + 0.43 * \text{economic} + 0.38 * \text{POS-ADJ} - 0.3 * \text{CAPITALIZATION} - 0.27 * \text{POS-VERB} + 0.27 * \text{regulatory} - 0.2 * \text{PAST-TENSE} + 0.18 * \text{biological} + 0.17 * \text{campaigner} + 0.17 * \text{POS-NUM} + 0.14 * \text{thermal} - 0.14 * \text{POS-ADV} - 0.12 * \text{POS-NOUN} + 0.1 * \text{health} - 0.1 * \text{C0} + 0.087 * \text{PARTICIPLE} + 0.087 * \text{POS-PROPN} - 0.084 * \text{PLURAL-NOUN} + 0.073 * \text{outdoor} + 0.055 * \text{chemical} + 0.0055 * \text{cultural}$$

$$\text{resident} = 0.54 * \text{citizens} + 0.49 * \text{native} + 0.37 * \text{visiting} - 0.19 * \text{PLURAL-NOUN} + 0.12 * \text{PAST-TENSE} + 0.11 * \text{caretaker} - 0.099 * \text{CAPITALIZATION} - 0.094 * \text{C0} + 0.082 * \text{PARTICIPLE} + 0.082 * \text{ward} + 0.077 * \text{POS-NOUN} - 0.039 * \text{POS-ADV} + 0.022 * \text{proprietor} + 0.022 * \text{POS-VERB} - 0.0065 * \text{POS-NUM} + 0.0045 * \text{POS-PROPN} + 0.0036 * \text{POS-ADJ}$$

$$
\begin{aligned}
\text{backers} &= 0.64 * \text{sponsors} \\
&+ 0.4 * \text{POS-NOUN} - 0.4 * \text{CAPITALIZATION} \\
&+ 0.33 * \text{advocates} + 0.28 * \text{PLURAL-NOUN} \\
&+ 0.19 * \text{POS-PROPN} + 0.18 * \text{businessman} \\
&+ 0.18 * \text{businessmen} + 0.16 * \text{fans} \\
&- 0.15 * \text{POS-ADJ} + 0.12 * \text{PARTICIPLE} \\
&+ 0.12 * \text{PAST-TENSE} - 0.12 * \text{POS-ADV} \\
&+ 0.092 * \text{whose} + 0.082 * \text{opposition} \\
&+ 0.065 * \text{POS-VERB} + 0.056 * \text{candidacy} \\
&+ 0.055 * \text{touted} + 0.047 * \text{startups} \\
&- 0.024 * \text{POS-NUM} + 0.024 * \text{rebels} \\
&+ 0.014 * \text{reformist} + 0.013 * \text{investment} \\
&- 0.002 * \text{C0}
\end{aligned}
$$

$$
\begin{aligned}
\text{rudimentary} &= 0.84 * \text{basics} \\
&- 0.65 * \text{C0} + 0.49 * \text{POS-ADJ} \\
&- 0.41 * \text{POS-VERB} + 0.41 * \text{apparatus} \\
&- 0.36 * \text{POS-NOUN} + 0.35 * \text{improvised} \\
&+ 0.15 * \text{POS-ADV} + 0.099 * \text{CAPITALIZATION} \\
&+ 0.072 * \text{PARTICIPLE} + 0.069 * \text{PLURAL-NOUN} \\
&+ 0.062 * \text{POS-NUM} + 0.059 * \text{develop} \\
&+ 0.05 * \text{PAST-TENSE} + 0.043 * \text{POS-PROPN}
\end{aligned}
$$

$$
\begin{aligned}
\text{admire} &= 0.73 * \text{admirable} \\
&- 0.66 * \text{PARTICIPLE} - 0.65 * \text{C0} \\
&+ 0.31 * \text{magnificent} + 0.23 * \text{CAPITALIZATION} \\
&+ 0.16 * \text{criticize} + 0.16 * \text{POS-NOUN} \\
&- 0.16 * \text{PAST-TENSE} + 0.14 * \text{loves} \\
&+ 0.1 * \text{POS-PROPN} + 0.1 * \text{beauty} \\
&- 0.098 * \text{POS-NUM} - 0.068 * \text{PLURAL-NOUN} \\
&+ 0.066 * \text{devotion} - 0.061 * \text{POS-ADV} \\
&- 0.058 * \text{POS-ADJ} + 0.039 * \text{openness} \\
&+ 0.02 * \text{charming} - 0.00015 * \text{POS-VERB}
\end{aligned}
$$

$$
\begin{aligned}
\text{re-add} &= -0.65 * \text{PARTICIPLE} \\
&- 0.47 * \text{POS-NUM} + 0.44 * \text{deleted} \\
&+ 0.43 * \text{POS-VERB} + 0.41 * \text{cruft} \\
&- 0.41 * \text{C0} - 0.3 * \text{PAST-TENSE} \\
&+ 0.28 * \text{section} + 0.19 * \text{CAPITALIZATION} \\
&+ 0.17 * \text{categorization} + 0.16 * \text{unblock} \\
&+ 0.15 * \text{POS-ADV} + 0.11 * \text{POS-PROPN} \\
&+ 0.098 * \text{reversion} - 0.09 * \text{POS-ADJ} \\
&+ 0.09 * \text{POS-NOUN} + 0.061 * \text{inserting} \\
&+ 0.046 * \text{reference} + 0.043 * \text{sourcing} \\
&+ 0.034 * \text{template} + 0.027 * \text{encyclopedic} \\
&+ 0.013 * \text{modify} - 0.0088 * \text{battleship} \\
&- 0.0071 * \text{cow} + 0.006 * \text{PLURAL-NOUN}
\end{aligned}
$$

$$
\begin{aligned}
\text{visuals} &= 0.47 * \text{cinematography} \\
&- 0.47 * \text{CAPITALIZATION} + 0.29 * \text{evocative} \\
&+ 0.25 * \text{multimedia} + 0.21 * \text{videos} \\
&+ 0.19 * \text{POS-NOUN} + 0.15 * \text{PLURAL-NOUN} \\
&+ 0.14 * \text{POS-PROPN} + 0.12 * \text{hallucinations} \\
&+ 0.11 * \text{awesome} + 0.1 * \text{PARTICIPLE} \\
&+ 0.08 * \text{video} - 0.079 * \text{POS-VERB} \\
&+ 0.076 * \text{sounds} + 0.076 * \text{POS-ADJ} \\
&+ 0.076 * \text{slick} + 0.075 * \text{POS-ADV} \\
&+ 0.066 * \text{C0} + 0.062 * \text{dazzling} \\
&+ 0.052 * \text{colorful} + 0.044 * \text{interactive} \\
&+ 0.027 * \text{jarring} + 0.019 * \text{visualization} \\
&+ 0.0047 * \text{PAST-TENSE} + 0.00025 * \text{POS-NUM}
\end{aligned}
$$

$$
\begin{aligned}
\text{hitter} &= -0.54 * \text{CAPITALIZATION} \\
&+ 0.45 * \text{C0} - 0.42 * \text{PLURAL-NOUN} \\
&+ 0.42 * \text{shortstop} + 0.36 * \text{designated} \\
&+ 0.32 * \text{batting} + 0.3 * \text{POS-VERB} \\
&+ 0.21 * \text{POS-NOUN} + 0.18 * \text{POS-ADV} \\
&+ 0.17 * \text{pitchers} + 0.17 * \text{pitcher} \\
&+ 0.14 * \text{catcher} - 0.12 * \text{PARTICIPLE} \\
&- 0.1 * \text{POS-NUM} - 0.096 * \text{inane} \\
&+ 0.087 * \text{guy} + 0.073 * \text{POS-PROPN} \\
&+ 0.048 * \text{exert} - 0.014 * \text{PAST-TENSE} \\
&+ 0.0071 * \text{outs} - 0.0064 * \text{POS-ADJ} \\
&+ 0.0019 * \text{swings}
\end{aligned}
$$

$$
\begin{aligned}
\text{Conflict} &= 0.61 * \text{POS-NOUN} \\
&- 0.49 * \text{POS-PROPN} + 0.44 * \text{warfare} \\
&+ 0.4 * \text{escalation} + 0.4 * \text{peace} \\
&+ 0.36 * \text{C0} + 0.24 * \text{guideline} \\
&+ 0.23 * \text{CAPITALIZATION} + 0.21 * \text{PARTICIPLE} \\
&+ 0.19 * \text{ethnic} - 0.19 * \text{POS-VERB} \\
&+ 0.16 * \text{resolved} + 0.12 * \text{POS-NUM} \\
&- 0.099 * \text{PLURAL-NOUN} + 0.078 * \text{PAST-TENSE} \\
&+ 0.07 * \text{divergence} + 0.065 * \text{geopolitical} \\
&- 0.05 * \text{stationary} + 0.038 * \text{POS-ADJ} \\
&- 0.032 * \text{shops} + 0.03 * \text{polarized} \\
&- 0.012 * \text{POS-ADV}
\end{aligned}
$$

$$
\begin{aligned}
\text{fence} &= 0.52 * \text{wire} \\
&+ 0.43 * \text{gates} - 0.41 * \text{CAPITALIZATION} \\
&+ 0.35 * \text{yard} - 0.32 * \text{PLURAL-NOUN} \\
&+ 0.21 * \text{shrubs} + 0.14 * \text{barn} \\
&+ 0.14 * \text{ditch} + 0.09 * \text{POS-VERB} \\
&+ 0.085 * \text{side} - 0.07 * \text{PARTICIPLE} \\
&- 0.052 * \text{POS-ADJ} + 0.042 * \text{POS-NUM} \\
&- 0.032 * \text{POS-PROPN} - 0.02 * \text{PAST-TENSE} \\
&- 0.012 * \text{C0} + 0.0068 * \text{nailed} \\
&- 0.0047 * \text{POS-ADV} + 0.00013 * \text{POS-NOUN}
\end{aligned}
$$

$$
\begin{aligned}
1978 = {} & 0.97 * \text{1970s} \\
& - 0.89 * \text{POS-ADJ} - 0.6 * \text{POS-PROPN} \\
& + 0.49 * \text{POS-NUM} - 0.42 * \text{POS-NOUN} \\
& + 0.21 * \text{C0} - 0.18 * \text{PLURAL-NOUN} \\
& - 0.12 * \text{POS-ADV} - 0.081 * \text{PARTICIPLE} \\
& - 0.073 * \text{CAPITALIZATION} + 0.067 * \text{POS-VERB} \\
& + 0.041 * \text{PAST-TENSE} + 0.039 * \text{seventies} \\
& + 0.026 * \text{contends}
\end{aligned}
$$

$$
\begin{aligned}
\text{heroine} = {} & 0.66 * \text{hero} \\
& + 0.35 * \text{protagonist} - 0.34 * \text{CAPITALIZATION} \\
& - 0.25 * \text{PLURAL-NOUN} + 0.14 * \text{actress} \\
& + 0.13 * \text{girl} + 0.1 * \text{C0} \\
& + 0.1 * \text{PAST-TENSE} + 0.071 * \text{POS-PROPN} \\
& + 0.07 * \text{POS-NOUN} + 0.063 * \text{POS-ADV} \\
& - 0.058 * \text{POS-NUM} + 0.051 * \text{protagonists} \\
& - 0.029 * \text{POS-VERB} + 0.026 * \text{PARTICIPLE} \\
& + 0.015 * \text{POS-ADJ} + 0.014 * \text{goddess}
\end{aligned}
$$

$$
\begin{aligned}
\text{structure} = {} & 0.91 * \text{structures} \\
& - 0.35 * \text{CAPITALIZATION} - 0.25 * \text{PLURAL-NOUN} \\
& + 0.17 * \text{structuring} + 0.16 * \text{POS-NOUN} \\
& - 0.085 * \text{POS-VERB} - 0.078 * \text{PAST-TENSE} \\
& + 0.05 * \text{POS-ADV} - 0.039 * \text{POS-ADJ} \\
& - 0.034 * \text{POS-PROPN} + 0.029 * \text{POS-NUM} \\
& + 0.026 * \text{structural} + 0.022 * \text{reorganization} \\
& - 0.022 * \text{C0} + 0.0079 * \text{PARTICIPLE}
\end{aligned}
$$

$$
\begin{aligned}
\text{wizards} = {} & 0.65 * \text{magic} \\
& + 0.41 * \text{witches} - 0.41 * \text{CAPITALIZATION} \\
& + 0.36 * \text{PLURAL-NOUN} + 0.19 * \text{POS-NOUN} \\
& - 0.19 * \text{POS-NUM} + 0.15 * \text{POS-ADJ} \\
& + 0.15 * \text{tech} + 0.13 * \text{POS-ADV} \\
& + 0.098 * \text{PAST-TENSE} + 0.09 * \text{wannabe} \\
& + 0.084 * \text{dragons} + 0.084 * \text{knights} \\
& + 0.052 * \text{C0} - 0.036 * \text{POS-PROPN} \\
& + 0.022 * \text{PARTICIPLE} + 0.015 * \text{err} \\
& - 0.013 * \text{POS-VERB} + 0.0041 * \text{guru}
\end{aligned}
$$

$$
\begin{aligned}
\text{autistic} = {} & 0.49 * \text{preschool} \\
& + 0.37 * \text{epilepsy} - 0.35 * \text{POS-NOUN} \\
& + 0.33 * \text{POS-ADJ} - 0.25 * \text{papal} \\
& + 0.22 * \text{son} + 0.21 * \text{POS-PROPN} \\
& + 0.16 * \text{twins} + 0.12 * \text{PAST-TENSE} \\
& + 0.12 * \text{therapist} + 0.12 * \text{PARTICIPLE} \\
& - 0.1 * \text{CAPITALIZATION} + 0.084 * \text{teenage} \\
& + 0.072 * \text{trait} + 0.069 * \text{psychologist} \\
& + 0.067 * \text{behaviors} + 0.056 * \text{kid} \\
& + 0.054 * \text{hospitalized} - 0.053 * \text{C0} \\
& + 0.052 * \text{manipulative} - 0.038 * \text{POS-NUM} \\
& + 0.037 * \text{granddaughter} - 0.03 * \text{rounds} \\
& + 0.03 * \text{PLURAL-NOUN} + 0.027 * \text{campers} \\
& + 0.026 * \text{POS-VERB} + 0.0092 * \text{dementia} \\
& - 0.0054 * \text{regional} - 0.0052 * \text{POS-ADV} \\
& - 0.0023 * \text{sedan}
\end{aligned}
$$

$$\begin{aligned}\text{tornado} &= 0.72 * \text{hurricane} \\ &+ 0.49 * \text{C0} - 0.47 * \text{CAPITALIZATION} \\ &+ 0.28 * \text{typhoon} - 0.26 * \text{PLURAL-NOUN} \\ &+ 0.23 * \text{tractor} + 0.21 * \text{POS-VERB} \\ &+ 0.16 * \text{POS-ADJ} + 0.12 * \text{POS-NUM} \\ &+ 0.1 * \text{flattened} - 0.1 * \text{ports} \\ &- 0.097 * \text{opium} + 0.072 * \text{POS-ADV} \\ &+ 0.053 * \text{POS-PROPN} + 0.053 * \text{avalanche} \\ &+ 0.052 * \text{tape} + 0.05 * \text{earthquake} \\ &- 0.045 * \text{colonial} - 0.043 * \text{POS-NOUN} \\ &+ 0.028 * \text{musical} - 0.025 * \text{handsets} \\ &+ 0.014 * \text{terrifying} + 0.013 * \text{PAST-TENSE} \\ &+ 0.013 * \text{occurrences} - 0.0067 * \text{labour} \\ &+ 0.0025 * \text{PARTICIPLE}\end{aligned}$$

$$\begin{aligned}1852 &= 0.85 * \text{1800s} \\ &- 0.81 * \text{POS-PROPN} - 0.78 * \text{POS-ADJ} \\ &- 0.61 * \text{POS-NOUN} + 0.45 * \text{POS-NUM} \\ &+ 0.43 * \text{C0} - 0.31 * \text{CAPITALIZATION} \\ &+ 0.29 * \text{renders} + 0.25 * \text{POS-VERB} \\ &- 0.19 * \text{PLURAL-NOUN} + 0.16 * \text{noted} \\ &- 0.15 * \text{PARTICIPLE} + 0.13 * \text{underscored} \\ &- 0.082 * \text{POS-ADV} + 0.063 * \text{insisting} \\ &+ 0.029 * \text{PAST-TENSE}\end{aligned}$$

$$\begin{aligned}\text{gloom} &= 0.66 * \text{gloomy} \\ &- 0.46 * \text{CAPITALIZATION} - 0.32 * \text{POS-VERB} \\ &+ 0.28 * \text{pessimism} + 0.28 * \text{darkness} \\ &+ 0.15 * \text{PAST-TENSE} - 0.14 * \text{PLURAL-NOUN} \\ &+ 0.14 * \text{POS-NOUN} + 0.12 * \text{PARTICIPLE} \\ &+ 0.11 * \text{POS-NUM} - 0.058 * \text{C0} \\ &- 0.058 * \text{POS-ADV} + 0.052 * \text{misery} \\ &+ 0.051 * \text{POS-PROPN} + 0.037 * \text{slump} \\ &- 0.025 * \text{POS-ADJ}\end{aligned}$$

$$\begin{aligned}\text{recycle} &= -0.65 * \text{PARTICIPLE} \\ &+ 0.61 * \text{bin} + 0.49 * \text{rubbish} \\ &- 0.48 * \text{C0} - 0.47 * \text{PAST-TENSE} \\ &+ 0.27 * \text{POS-VERB} + 0.18 * \text{POS-NOUN} \\ &+ 0.15 * \text{utilize} + 0.14 * \text{plastic} \\ &- 0.12 * \text{POS-NUM} + 0.1 * \text{excess} \\ &+ 0.088 * \text{sustainability} + 0.083 * \text{POS-PROPN} \\ &+ 0.066 * \text{aluminum} + 0.042 * \text{synthesize} \\ &- 0.037 * \text{PLURAL-NOUN} + 0.036 * \text{POS-ADJ} \\ &+ 0.035 * \text{refurbished} + 0.034 * \text{POS-ADV} \\ &+ 0.03 * \text{converter} + 0.026 * \text{nitrogen} \\ &- 0.004 * \text{CAPITALIZATION} + 0.0024 * \text{saving}\end{aligned}$$