Game-theoretic Vocabulary Selection via the Shapley Value and Banzhaf Index

Roma Patel [*]
Brown University

Marta Garnelo
DeepMind

Ian Gemp
DeepMind

Chris Dyer
DeepMind

Yoram Bachrach
DeepMind

Abstract

The input vocabulary and their learned representations are crucial to the performance of neural NLP models. Using the full vocabulary results in less explainable and more memory intensive models, with the embedding layer often constituting the majority of model parameters. It is thus common to use a smaller vocabulary to lower memory requirements and construct more interpertable models.

We propose a vocabulary selection method that views words as members of a team trying to maximize the model's performance. We apply power indices from cooperative game theory, including the Shapley value and Banzhaf index, that measure the relative importance of individual team members in accomplishing a joint task. We approximately compute these indices to identify the most influential words.

Our empirical evaluation examines multiple NLP tasks, including sentence and document classification, question answering and textual entailment. We compare to baselines that select words based on frequency, TF-IDF and regression coefficients under L1 regularization, and show that this game-theoretic vocabulary selection outperforms all baselines on a range of different tasks and datasets.

1 Introduction

Most state-of-the-art NLP methods use neural networks that require a pre-defined vocabulary to vectorise and encode text. In large text datasets, the vocabulary size can grow to hundreds of thousands of words, and having an embedding space over the entire vocabulary results in models that are expensive in memory and compute, and hard to interpret.

Many of the words in the vocabulary are not crucial to task performance, and can be removed without a significant drop in final task performance.

superbly photographed and staged by mendes
with a series of riveting set pieces the
likes of which mainstream audiences have
rarely seen .

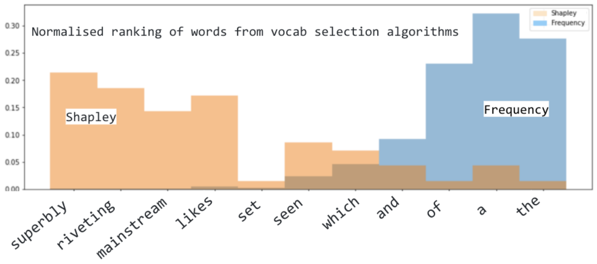


Figure 1: An example sentence from SST-2 (Socher et al., 2013), as well as the distribution of heuristic values based on vocabulary selection algorithms. Frequency and TF-IDF weight stopwords (right) higher whereas a game-theoretic Shapley-based approach tends to value task-specific words (left) more.

It is common to use heuristics such as frequency or TF-IDF to reduce vocabulary size. After filtering to obtain a smaller vocabulary, "out-of-vocabulary" (OOV) words are replaced with an unknown word token <UNK>. This reduction in vocabulary size has many advantages. Models with reduced vocabulary are more easily interpretable and achieve increased transparency (Adadi and Berrada, 2018; Samek et al., 2019), require less memory, can be used in resource constrained settings, and are less prone to overfitting (Sennrich et al., 2015; Shi and Knight, 2017; L'Hostis et al., 2016; Chen et al., 2019). However, reducing the vocabulary size with a heuristic such as frequency is often not optimal. For example, Figure 1 shows the top ranked words according to frequency (blue), that are largely unimportant for the sentiment task at hand.

We consider the **vocabulary selection problem:** given a target vocabulary size k (or equivalently, a target memory footprint or a "budget" of model parameters for the embedding layer), what is the optimal word subset we should use as our vocabulary? Our solution's output, based on the Shapley

[*] Work done during an internship at DeepMind. Corresponding email: romapatel@brown.edu or yorambac@google.com.

Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 2789–2798
June 6–11, 2021. ©2021 Association for Computational Linguistics

value, is also shown in Figure 1, demonstrating that it focuses on words relevant to the task.

Our Contribution: We use game theoretic principles to propose a vocabulary selection method. We cast the vocabulary selection problem as a *cooperative game*, which considers subset of words as a "team" whose goal is to solve the NLP task at hand. We define the performance of a team as the performance of a model that uses only those words as its vocabulary. Our method applies solution concepts from game theory to determine the relative importance of each word in achieving the goal. Specifically, we consider the Shapley value (Shapley, 1953) and Banzhaf index (Banzhaf III, 1964), key concepts in game theory, that are used as "power-indices" for measuring the individual contribution of team members to the success of the team. We approximate these indices by sampling subsets of words and training a model on each subset to contrast model performance when including and omitting a target word.

We evaluate our approach against baselines such as TF, TF-IDF and ranking using logistic regression coefficients under L1 regularization. We evaluate on a range of datasets and task structures: single-sentence classification, pairwise-sentence classification and document classification. While our method is *significantly more demanding computationally* than these simple baselines, we empirically demonstrate that it outperforms these baselines on all tasks, offering better tradeoffs between the vocabulary size and the model's performance.

2 Method

We assume a dataset D and a training method M for training on D and producing a model f_θ where θ are tuned model parameters. The model is evaluated on a validation set T to estimate how well the model generalizes to the true data distribution. An evaluation metric (for example the model accuracy or F_1 score, as evaluated on the validation set) for each model f_θ is denoted by $q(f_\theta)$, thus allowing an assessment of the performance of a subset of words. We first briefly discuss preliminaries from cooperative game theory (Chalkiadakis et al., 2011).

2.1 Preliminaries: Cooperative Game Theory

Cooperative game theory investigates settings where multiple players work together in teams. A (transferable-utility) cooperative game consists of a set $A = \{a_1, \ldots, a_n\}$ of players and a characteristic function $v : 2^A \to \mathbb{R}$ mapping any subset of players $C \subseteq A$ (called a "team" or "coalition") to a real value $v(C)$ indicating the performance of the team when working together.

The **Shapley value** (Shapley, 1953), denoted $\phi(v) = (\phi_1, \ldots \phi_n)$, reflects each player's individual contribution to the success of the team, adhering to fairness axioms (Dubey, 1975). [1] Similarly, the **Banzhaf index** (Banzhaf III, 1964), denoted $\beta(v) = (\beta_1, \ldots \beta_n)$, measures impact of individuals on the success of a team, using different axioms (Dubey and Shapley, 1979; Strafiin Jr, 1988).

Consider quantifying the individual contribution of a player $a_i \in A$ in a game with the characteristic function v. Examine the player a_i and a coalition $C \subseteq A \setminus \{a_i\}$ that does not contain that player. The **marginal contribution** of a_i to the coalition C is defined as $m(a_i, C) = v(C \cup \{a_i\}) - v(C)$, i.e. the increase in value arising from adding a_i to the coalition C. Similarly, denote the set of *permutations* over then n players as Π (i.e. each $\pi \in \Pi$ is a bijection $\pi : A \to A$), and denote the predecessors of $a_i \in A$ in the permutation π as $b(a_i, \pi)$. The marginal contribution of a_i in the permutation π is defined as $m(a_i, \pi) = v(b(a_i, \pi) \cup \{a_i\}) - v(b(a_i, \pi))$, i.e. the increase in value arising from adding a_i to the players appearing before it in the permutation π.

The Banzhaf index β_i of player a_i is the marginal contribution of player a_i averaged over all possible coalitions that do not contain that player:

$$\beta_i = \frac{1}{2^{n-1}} \sum_{C \subset A | i \in C} v(C \cup \{i\}) - v(C)$$

The Shapley value π_i of a player a_i is the marginal contribution of that player, averaged across all permutations:

$$\phi_i = \frac{1}{n!} \sum_{\pi \in \Pi} v(b(a_i, \pi) \cup \{a_i\}) - v(b(a_i, \pi))$$

The Banzhaf index of a_i can be viewed as the expected increase in performance under uncertainty about the participation of other players in the team

[1] The Shapley value has also been used to examine power in team formation (Aziz et al., 2009; Mash et al., 2017; Bachrach et al., 2020), combinatorial tasks (Ueda et al., 2011; Banarse et al., 2019), pricing and auctions (Bachrach, 2010; Kamboj et al., 2011; Blocq et al., 2014) or political settings (Bilbao et al., 2002; Bachrach et al., 2011; Filmus et al., 2019), or feature importance for model explainability (Lundberg and Lee, 2017).

— if each of the other players has an equal probability of joining the team or not joining it, how much value to we expect to add when a_i joins the team. Similarly, the Shapley value can be viewed as the expected increase in team value that a_i would yield when players join the team in a random order. [2]

2.2 Our Approach: Vocabulary Selection by Comparing Power Indices

Given the entire vocabulary V and a budget of k words to use, our method selects a subset $V' \subset V$ where $|V'| = k$, optimizing the performance $q(f_\theta^{V'})$ of a model $f_\theta^{V'}$ trained using a vocabulary consisting only of the words in V'.

We view each word as a player and each subset of words $C \subseteq V$ as a team, and construct a cooperative game. The characteristic function $v : V \to \mathbb{R}$ maps a subset of words (partial vocabularies) to the performance obtained when training a model with only these words a vocabulary. Formally, we define the performance $v(C)$ of the team $C \subseteq V$ to be the performance $q(f_\theta^C)$ of an NLP model f_θ^C with the words in C as its input vocabulary. [3]

Given a vocabulary $C \subseteq V$, evaluating $v(C)$ requires training a model f_θ on dataset D using only the words in C as the vocabulary [4], and measuring its performance on the validation set T to obtain $v(C) = q(f_\theta^C)$. We compute the Shapley value ϕ_i or Banzhaf index β_i of any word $w_i \in V$ (see Section 2.1). Words with high values are ones that have a larger positive influence on performance, whereas words with lower values are ones that do not impact task performance when they are removed. [5]

Observe that the Banzhaf index β_i is the expected marginal contribution $m(a_i, C)$ for a coalition C sampled uniformly at random from the set $\{C \subseteq V | a_i \in C\}$, and the Shapley value ϕ_i is the expected marginal contribution $m(a_i, \pi)$ for a permutation π sampled uniformly at random from Π. We can approximate these by taking a sample of coalitions or permutations, and examining a_i's average marginal contribution in the sample. For the Shapley value, the sample consists of permutations of words in the vocabulary, where for each permutation π we train two models on vocabularies that differ by a single word w. The performance difference between the two models is then the marginal contribution of the word w. For the Banzhaf index, we directly construct the vocabulary by flipping a fair coin per word to determine its inclusion in the vocabulary. The power index is approximated as the average marginal contribution of the word across the samples. Finally, we select the k words with the highest power index as our vocabulary V'. This is shown in Algorithms 1, 2.

Algorithm 1 Banzhaf Vocabulary Selection

```
Inputs: NLP dataset $D$ with full vocabulary $V$
for each word $w$ in $V$ do
    $\beta_w \leftarrow 0$ (initialise Banzhaf index estimate)
    for i=1 to $S$ (number of sampled coalitions) do
        $C_1 \leftarrow \emptyset$
        for j=1 to $|V|$ do
            $s \leftarrow$ Uniform($\{0, 1\}$))
            if $s = 1$ and $w_j \neq w$ then
                $C_1 \leftarrow C_1 \cup \{w_j\}$
            end if
        end for
        $C_2 \leftarrow C_1 \cup \{w\}$ (random coalition including $w$)
        $f_\theta^{C_1} \leftarrow$ TrainModel($C_1$) (Train on vocabulary $C_1$)
        $f_\theta^{C_2} \leftarrow$ TrainModel($C_2$) (Train on vocabulary $C_2$)
        $m(w, C_1) \leftarrow q(f_\theta^{C_2}) - q(f_\theta^{C_1})$
        $\beta_w \leftarrow \beta_w + m(w, C_1)$
    end for
    $\beta_w \leftarrow \frac{1}{S}\beta_w$ (average marginal contributions)
end for
Rank words in $V$ based on Banzhaf estimates $\beta_w$
Return top $k$ words in ranking
```

Algorithm 2 Shapley Vocabulary Selection

```
Inputs: NLP dataset $D$ with full vocabulary $V$
for each word $w$ in $V$ do
    $\phi_w \leftarrow 0$ (initialise Shapley value estimate)
    for i=1 to $S$ (number of sampled permutations do
        $\pi \leftarrow$ Random-Permutation($V$)
        $C_1 \leftarrow b(w, \pi)$ (predecessors of $w$)
        $C_2 \leftarrow C_1 \cup \{w\}$ (predecessors including $w$)
        $f_\theta^{C_1} \leftarrow$ TrainModel($C_1$) (Train on vocabulary $C_1$)
        $f_\theta^{C_2} \leftarrow$ TrainModel($C_1$) (Train on vocabulary $C_2$)
        $m(w, \pi) \leftarrow q(f_\theta^{C_2}) - q(f_\theta^{C_1})$
        $\phi_w \leftarrow \phi_w + m(w, \pi)$
    end for
    $\phi_w \leftarrow \frac{1}{S}\phi_w$ (average marginal contributions)
end for
Rank words in $V$ based on Shapley estimates $\pi_w$
Return top $k$ words in ranking
```

3 Evaluation

We evaluate our algorithm on multiple tasks, contrasting its performance with common baselines.

[2] An equivalent formula for the Shapley value is: $\phi_i = \sum_{C \subset A | i \in C} \frac{|C|!(|A|-|C|-1)!}{|A|!} v(C \cup \{i\}) - v(C)$, showing the different weights the indices give to different size coalitions.

[3] For example, for text classification we may define $v(C)$ to be the model's accuracy when using C as the vocabulary.

[4] For example in a text classification task, one could train a neural network classifier f_θ^C on the dataset D, replacing all the words in $V \setminus C$ with the UNK token.

[5] The direct formulas for the Shapley or Banzhaf indices enumerate over *all* possible word subsets or permutations, which is intractable. Hence, we use an approximation algorithm (Matsui and Matsui, 2000; Bachrach et al., 2010).

3.1 Datasets and Tasks

We consider three different task structures.

Single Sentence Classification: the task requires a model to encode the words of a given sentence and output a classification based on properties of sentences (for e.g., sentiment or acceptability). We evaluate on a sentiment-analysis task using the SST-2 dataset (Socher et al., 2013) and a corpus acceptability task using the CoLA dataset (Warstadt et al., 2019; Wang et al., 2018). The sentiment analysis task contains 9.6k sentences labelled with a positive or negative sentiment, while the acceptability task contains 8.5k sentences labelled with an acceptability judgement about whether or not it is a grammatically correct English sentence.

Entailment and Question Pair Classification: this task requires a model to encode two sentences and output a classification based on the relation between them. We evaluate on a textual entailment task using the SNLI dataset (Bowman et al., 2015a) and a question pair classification task using the QQP dataset (Wang et al., 2018). SNLI contains 550k sentence pairs and requires models to encode two different sentences, a premise and a hypothesis, and predict one of three relations between them: an entailment, a contradiction or a neutral relation. The QQP task contains 364k pairs and requires models to encode two different text inputs, a question and an alternate question composed of different words, and to predict whether or not the two questions correspond to the same answer.

Document Classification: this task requires models to encode an input document or article, and predict a class based on properties of the document. We evaluate on the AG-News and Yelp datasets (Zhang et al., 2015). The AG-News dataset contains the title and description of 120,000 news articles in four categories (the prediction target is the category). The Yelp dataset contains 130,000 million samples with text reviews, with the prediction target being the polarity of the review (positive or negative). The number of words in each text instance (document) are significantly larger than in the single sentence classification task, requiring models to capture phenomena like co-reference and temporal order that are prevalent in longer texts.

3.2 Methodology

Our method in Section 2.2 is agnostic to the specific model and training procedure: we simply assume we have access to an algorithm that trains on a dataset D and produces a trained model f_θ whose quality $q(f_\theta)$ is evaluated on a validation set T.

We perform our empirical evaluation using both an LSTM classifier and a logistic regression classifier. Our method trains many models with different vocabularies to select the final vocabulary V'. We then evaluate the quality of the chosen reduced vocabulary V' by training a final model $f^{V'}$ which uses only the vocabulary V' and evaluate the performance of $f^{V'}$ on a held out test T'.

To maximize performance, one should use the same architecture during the vocabulary selection process as the evaluation. However, words that are strong features for one architecture are likely to also be strong features for another architecture. Hence, we can select the the vocabulary using one architecture even if we intend to use this vocabulary for another architecture. As our vocabulary selection procedure trains many models, we use logistic regression models during the vocabulary selection process. We show it still significantly outperforms baselines, and allows faster and more efficient computation of the Shapley value. We then evaluate the quality of the vocabulary using an LSTM model.

Training logistic regression models: To train the logistic regression classifier in the single-text case, we represent each sentence or document as the set of words that occur in that text sample. For the pairwise-sentence case, we similarly represent each paired input with *three times* the number of word features, using a one hot encoding indicating that the word occurred only in the first sentence (e.g. question), only in the second sentence (e.g. answer) or whether it occurred in both sentences. This model is far simpler than state-of-the-art text classification models, but we find it is a good-enough proxy for the Shapley computation step, and much more economical computationally.

Evaluating the Selected Vocabulary's Quality

To train the LSTM classifier, we encode words using an embedding layer of size 100. These embeddings are fed one at a time to an LSTM encoder with a hidden layer size of 100, and the output of the LSTM encoder is fed into a feedforward neural network yielding the final classification (Deng and Liu, 2018) over some number of classes.

Our experiments show that even when using the simple logistic regression for the vocabulary selection process we achieve a significant performance improvement over baselines, as evaluated with an LSTM model. In other words, the vocabulary quality improvement transfers to more complex models.

Baselines

We contrast the performance of our approach (Algorithm 1 based on the Banzhaf index and Algorithm 2 based on the Shapley value) with several baselines. We first consider ranking by term frequency (TF), i.e selecting the most frequently occuring words in the dataset. We also consider ranking words by TF-IDF scores (Ramos et al., 2003), which is commonly used for web search. As a stronger baseline we consider ranking words based on their regression coefficients, a method used for estimating feature importance (Ellis, 2010; Nimon and Oswald, 2013). In this baseline, we train a logistic regression model with L_1 regularization on the dataset D (the regularization encourages the model to have low weights, setting the weight of many features to zero when the regularization is strong enough); we then rank features by the absolute coefficient of each feature in the trained model. We refer to this as the L_1 **baseline.** [6]

Our approach for calculating the Banzhaf index or Shapley value is based on a random sample of coalitions, and achieving a good accuracy requires taking many samples, especially when ranking a vocabulary with many words. To keep the required compute manageable while achieving a reasonable approximation, we first apply a **pre-filtering** step, selecting a large vocabulary (but not the full vocabulary) by applying the TF heuristic, then selecting the final small vocabulary from this large vocabulary using our approach. For instance, with a target vocabulary size of 100 words, we first filter out all but the 1,000 most frequent words and then rank based on the Shapley value (and contrast the performance of this method with selecting the top 100 words based solely on TF or TF-IDF score). When comparing against the L_1 baseline, we similarly apply an L_1 based pre-filtering.

4 Empirical Results

We analyze the performance of our method and the baselines across a range of target vocabulary sizes, investigating which method achieves a better trade-off between vocabulary size and model quality.

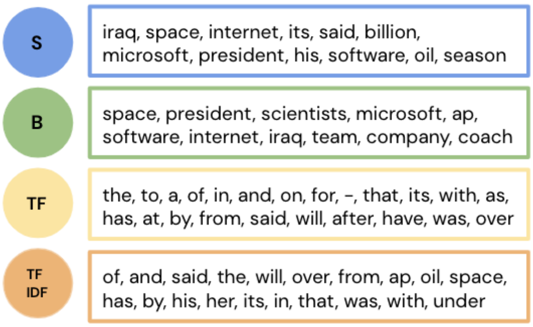

S
iraq, space, internet, its, said, billion, microsoft, president, his, software, oil, season
B
space, president, scientists, microsoft, ap, software, internet, iraq, team, company, coach
TF
the, to, a, of, in, and, on, for, –, that, its, with, as, has, at, by, from, said, will, after, have, was, over
TF IDF
of, and, said, the, will, over, from, ap, oil, space, has, by, his, her, its, in, that, was, with, under

Figure 2: Example of top ranked words on the AG-News based on various methods: Shapley (**S**), Banzhaf (**B**), **TF** and **TF-IDF**.

Vocabulary size and model quality tradeoffs

Figure 3 contrasts our method with the TF and TF-IDF baselines in the SST-2 dataset. It shows that for all methods, increasing the allowed vocabulary size improves the model quality (at the cost of an increased number of parameters or memory).

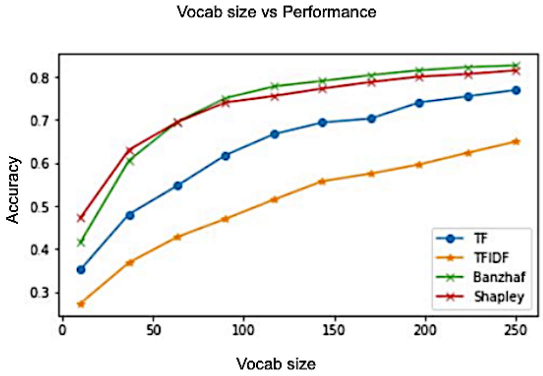
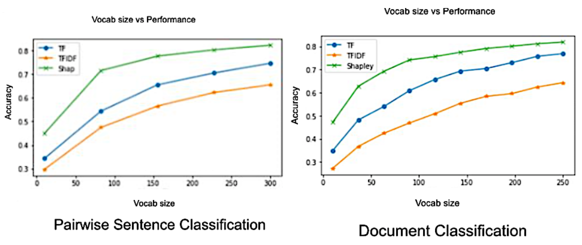


Figure 3: Performance of Shapley (red), Banzhaf (green), TF (blue) and TF-IDF (orange) on AG-News.

The figure indicates that both the Banzhaf and Shapley algorithms offer a significantly better trade-off between vocabulary size and model quality — they produce a better performing model at all the tested vocabulary sizes (the performance gap is especially pronounced for smaller vocabulary sizes).

Interestingly, the performance of both the Banzhaf and Shapley is very similar. Although they both select words with high marginal contributions, they rely on different power indices. To determine whether they select the same words, we examined the words selected at a target vocabulary size of $|V'| = 100$. Figure 2 shows the top words according to the different methods. The top 100

[6] In logistic regression with L_1 regularization, the regression coefficients and derived word ranking depend on the degree of regularization and the initialization. Methods like GLMpath (Friedman et al., 2010) obtain the entire L_1 path of the GLM at the cost of fitting a single model. In the spirit of stability selection (Meinshausen and Bühlmann, 2010), to alleviate stochasticity we average 20 training runs of the L_1-regularized model, averaging coefficients to obtain the ranking over words (still cheaper computationally than our approach).



Pairwise Sentence Classification

Document Classification

Figure 4: Performance of a Shapley, TF and TF-IDF in additional task structures.

words under the Banzhaf and Shapley algorithms intersected on less than 70% of the words, so although they have similar performance, there are non-negligible differences in the words they select.

Figure 3 relates to single sentence classification. Figure 4 shows similar results for the two other types of tasks: pairwise sentence classification and document classification. Similarly to the previous figure, these results indicate that our approach achieves a significantly better tradeoff between vocabulary size and model accuracy. This indicates that our proposed approach offers advantages across a wide set of NLP tasks.

Table 1 shows the performance of an LSTM classifier across all tasks and datasets for the various methods. It shows a consistent improvement over the baselines in all the tasks for both the Banzhaf and Shapley methods (which have very similar performance in all the datasets).

Comparison with the L_1 baseline: Section 3.2 considered the stronger baseline of ranking by regression coefficients in an L_1 regularized logistic regression. The high-level motivation of this baseline is similar to our approach in that words are ranked based on their influence as measured by training a model; however, the L_1 method trains a single model (or has a computational cost similar to training one or few models), whereas a power index computation relies on training a *sample* of models. Figure 5 shows our approach outperforms the L_1 baseline.

Task & Dataset	Method	Vocab	Acc
SST-2	TF-IDF	17,539	80.2
(Socher et al., 2013)	Frequency		80.3
	Banzhaf		81.7
	Shapley		**81.9**
COLA	TF-IDF	9007	63.5
(Warstadt et al., 2019)	Frequency		63.7
	Banzhaf		63.9
	Shapley		**64.2**
SNLI	TF-IDF	42,392	83.9
(Bowman et al., 2015b)	Frequency		83.9
	Banzhaf		84.1
	Shapley		**84.3**
QQP	TF-IDF	117,303	80.8
(Wang et al., 2018)	Frequency		81.2
	Banzhaf		**81.9**
	Shapley		**81.9**
AG-NEWS	TF-IDF	159,697	79.6
(Zhang et al., 2015)	Frequency		78.5
	Banzhaf		79.9
	Shapley		**80.2**
YELP	TF-IDF	458,705	84.5
(Zhang et al., 2015)	Frequency		83.9
	Banzhaf		86.7
	Shapley		**87**

Table 1: Performance of vocabulary selection methods across datasets and tasks, at a target vocabulary size of $|V'| = 750$ words (column 3 is initial vocabulary size). Note performance is lower than state-of-the-art methods, as results are based on a significantly reduced vocabulary size (and using a simple LSTM architecture, with no hyperparameter tuning).

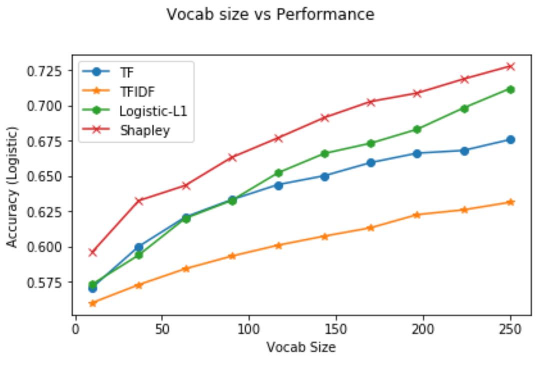


Figure 5: Performance on SST-2 of Shapley (red), **Logistic-L1** (green), TF (blue) and TF-IDF (orange).

Comparison with subword approaches: Subword embeddings (Sennrich et al., 2015) is a recent approach which considers tokens that can be parts of words, resulting in a less sparse vocabulary and having features shared across words. Such approaches are flexible and allow choosing a target vocabulary size. Our approach can also work with subword embeddings: after computing some set of subwords over the vocabulary, we can still filter out less important subwords to improve task performance. We evaluated whether applying our approach on top of using subword embeddings can still lead to improved performance. We first run a byte-pair encoding (BPE) algorithm (Sennrich et al., 2015; Provilkov et al., 2019; Kudo and Richardson, 2018) over each input vocabulary for a dataset. This algorithm operates by merging together the most frequent sequence of adjacent tokens in each iteration. We do this for a total number of 10,000 merges, resulting in a smaller vocabulary that now composed of subwords. We then apply Shapley, Banzhaf, TF and TF-IDF rankings of these subword tokens, as we have done in the word-level experiments. Figure 6 shows that we have improved performance over the baselines in the subword case as well.

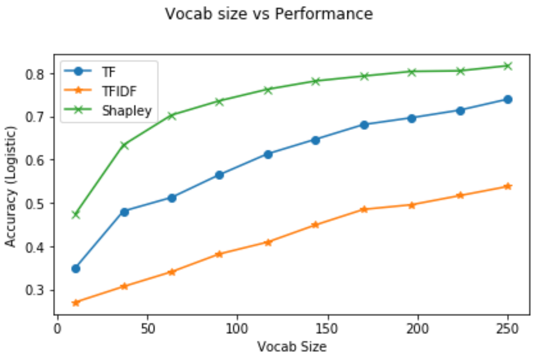


Figure 6: Performance of Shapley (green), TF (blue) and TF-IDF (orange) when considering **subword embeddings** on the AG-News dataset.

5 Discussion

The results in Section 4 show that a game theoretic approach to vocabulary selection can achieve better tradeoffs between the vocabulary size and model performance than heuristics such as TF and TF-IDF based selection, or a method based on regression coefficients in an L_1 regularized logistic regression. This advantage comes at the cost of having a significantly higher computational cost of selecting the vocabulary. Following the expensive selection step, we now have the benefit of a smaller model which is more interpretable and explainable, has a reduced memory consumption and potentially less prone to overfitting. We have proposed several ways to mitigate the compute load of selecting the vocabulary: applying a heuristic pre-filtering step and using logisitic regression models rather than the full model while estimating power indices.

6 Related Work

We proposed a vocabulary selection method for NLP tasks, using cooperative game theory. We discuss related work on model compression, tailoring the vocabulary in NLP tasks and using subword embeddings, and approximating game theoretic solutions and using them for explainable AI.

Model compression: Using the full vocabulary to train models limits the applicability of models in memory-constrained or computation-constrained scenarios (Faruqui et al., 2015; Yogatama et al., 2015). Earlier work discusses methods for compressing model size. These yield models that are less expensive in memory and compute, and that are also more easily interpretable. Model compression methods include matrix compression methods such as sparsification of weights in a matrix (Wen et al., 2016), Bayesian inference for compression (Molchanov et al., 2017; Neklyudov et al., 2017), feature selection methods such as ANOVA (Girden, 1992), precision reduction methods (Han et al., 2015; Hubara et al., 2017) and approximations of the weight matrix (Tjandra et al., 2017; Le et al., 2015). Our method relies on game theoretic principles; it filters our vocabulary words, and can thus operate with any NLP architecture (i.e. the method is agnostic to the model architecture used). Further, the interpretability in our case stems from having few features, clearly highlighting the most impactful features in the dataset.

Vocabulary selection methods and subword and character level embeddings: earlier work examined selecting a vocabulary for an NLP task. Some alternatives drop out words (Chen et al., 2019), whereas character-level methods that attempt to represent the input text at the level of individual characters (Kim et al., 2015; Lee et al., 2017; Ling et al., 2015) while subword methods attempt to tokenize words into parts of words in a more efficient way (Sennrich et al., 2015; Kudo and Richardson, 2018).

Character level embedding methods decompose words to allow each individual character to have its own embedding. This reduces the vocabulary size to the number of characters, much smaller than the number of words in the full vocabulary. However, this is not applicable for some character-free languages (e.g. Chinese, Japanese, Korean). Also, such methods have reduced performance, and typically use larger embedding sizes than word embedding models to obtain reasonable quality (Zhang et al., 2015; Kim et al., 2015).

In contrast, **subword embeddings** have shown improved performance for several NLP tasks. Such methods typically merge pairs of frequent character sequences, to get a more optimal token vocabulary from an information-theoretic viewpoint. Byte-pair encoding (BPE) algorithms construct subword vocabulary that is less sparse, and increases shared features between words [7], allowing better propogation of semantic meaning. As shown in Section 4, our method can operate on top of subword embeddings, and achieve good tradeoffs between the model size and performance.

Cooperative game theory and applications for explainable AI: we use concepts from game theory, viewing words as players in a game whose goal is to improve model performance. Such settings have been a key topic of study in game theory since the 1950s (Weintraub, 1992). Many solution concepts have been proposed, examining issues such as stability and fairness. Power indices such as the Banzhaf index (Banzhaf III, 1964) and Shapley value (Shapley, 1953) to measure the relative impact players have on the outcome of the game. It is computationally hard to calculate them even in simple games (Matsui and Matsui, 2001; Elkind et al., 2007). We have applied a Monte-Carlo sampling approximation based on existing methods (Fatima et al., 2008; Bachrach et al., 2010).

Our use of the Shapley value is akin to recent explainable AI methods, that attempt to allow AI models to provide human readable insights to explain their decisions (Adadi and Berrada, 2018; Samek et al., 2019). For example, power indices (such as the Shapley value) have been used to explain individual model predictions (Datta et al., 2016; Lundberg and Lee, 2017), by estimating the contribution of individual features on each prediction. This can be done for linear models (Lundberg and Lee, 2017) as well as tree-based models (Lundberg et al., 2020).

Explainable AI methods typically take a trained model and a given instance as input, and perturb the features of the instance, using the same model to output predictions for many perturbed inputs. In contrast, our goal is not to understand the predictions of a given model, but to select an small input vocabulary set for a task, focusing on the most relevant part of the input space and yielding simpler and more interpretable models. Further, we train many models to estimate contributions, rather than perturbing the inputs for a single model.

7 Conclusion

We proposed a vocabulary selection method based on cooperative game theory and empirically showed improvements over baselines in multiple NLP tasks. Our approach, with its task-specific vocabulary, offers an improved model size and quality tradeoffs.

Several questions remain open for future research on better vocabulary selection. Could alternative power indices, apart from what we have shown using the Shapley and Banzhaf indeces, achieve better performance? Is there a way to better combine our methods with subword embeddings? Moreover, given that our method is computationally demanding during vocabulary construction time, an interesting problem is to explore ways to speed up this process; both theoretically, through a different power index calculation, and practically, through better parallelization.

8 Acknowledgements

We would like to acknowledge Thore Graepel for formative advice and discussions at early stages of the project. We would also like to thank Richard Everett, Thomas Anthony, Andrea Tacchetti and Tom Eccles for helpful questions and comments throughout the project, and Ellie Pavlick for suggestions on robustly evaluating NLI models. We would also like to acknowledge the anonymous reviewers and area chairs, whose feedback and suggested changes for additional experiments were hugely helpful in making the paper stronger, more robust, and clearer than before.

References

Amina Adadi and Mohammed Berrada. 2018. Peeking inside the black-box: A survey on explainable artificial intelligence (xai). *IEEE Access*, 6:52138–52160.

Haris Aziz, Oded Lachish, Mike Paterson, and Rahul Savani. 2009. Power indices in spanning connectivity games. In *International Conference on Algorithmic Applications in Management*, pages 55–67. Springer.

Yoram Bachrach. 2010. Honor among thieves: collusion in multi-unit auctions. In *AAMAS*.

Yoram Bachrach, Edith Elkind, and Piotr Faliszewski. 2011. Coalitional voting manipulation: A game-theoretic perspective. In *IJCAI*.

[7] For instance, the word *"sadder"* could be split into *"sad"* and *"er"*, where the ending *"er"* has a similar meaning in other circumstances — *"faster"*, *"nearer"* etc.

Yoram Bachrach, Richard Everett, Edward Hughes, Angeliki Lazaridou, Joel Z Leibo, Marc Lanctot, Michael Johanson, Wojciech M Czarnecki, and Thore Graepel. 2020. Negotiating team formation using deep reinforcement learning. *AIJ*, 288.

Yoram Bachrach, Evangelos Markakis, Ezra Resnick, Ariel D Procaccia, Jeffrey S Rosenschein, and Amin Saberi. 2010. Approximating power indices: theoretical and empirical analysis. *JAAMAS*, 20(2):105–122.

Dylan Banarse, Yoram Bachrach, Siqi Liu, Guy Lever, Nicolas Heess, Chrisantha Fernando, Pushmeet Kohli, and Thore Graepel. 2019. The body is not a given: Joint agent policy learning and morphology evolution. In *AAMAS*.

John F Banzhaf III. 1964. Weighted voting doesn't work: A mathematical analysis. *Rutgers L. Rev.*, 19:317.

Jesús Mario Bilbao, Julio R Fernandez, Nieves Jiménez, and Jorge J Lopez. 2002. Voting power in the european union enlargement. *EJOR*, 143(1):181–196.

Gideon Blocq, Yoram Bachrach, and Peter Key. 2014. The shared assignment game and applications to pricing in cloud computing. In *AAMAS*.

Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015a. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326*.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015b. A large annotated corpus for learning natural language inference. In *EMNLP*.

Georgios Chalkiadakis, Edith Elkind, and Michael Wooldridge. 2011. Computational aspects of cooperative game theory. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 5(6):1–168.

Wenhu Chen, Yu Su, Yilin Shen, Zhiyu Chen, Xifeng Yan, and William Wang. 2019. How large a vocabulary does text classification need? a variational approach to vocabulary selection. *arXiv preprint arXiv:1902.10339*.

Anupam Datta, Shayak Sen, and Yair Zick. 2016. Algorithmic transparency via quantitative input influence: Theory and experiments with learning systems. In *IEEE SP*, pages 598–617.

Li Deng and Yang Liu. 2018. *Deep learning in natural language processing*. Springer.

Pradeep Dubey. 1975. On the uniqueness of the shapley value. *IJGT*, 4(3):131–139.

Pradeep Dubey and Lloyd S Shapley. 1979. Mathematical properties of the banzhaf power index. *MOR*, 4(2):99–131.

Edith Elkind, Leslie Ann Goldberg, Paul Goldberg, and Michael Wooldridge. 2007. Computational complexity of weighted threshold games. In *AAAI*.

Paul D Ellis. 2010. *The essential guide to effect sizes: Statistical power, meta-analysis, and the interpretation of research results*. Cambridge University Press.

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

Shaheen S Fatima, Michael Wooldridge, and Nicholas R Jennings. 2008. A linear approximation method for the shapley value. *AIJ*, 172(14):1673–1699.

Yuval Filmus, Joel Oren, Yair Zick, and Yoram Bachrach. 2019. Analyzing power in weighted voting games with super-increasing weights. *TCS*, 63(1):150–174.

Jerome Friedman, Trevor Hastie, and Rob Tibshirani. 2010. Regularization paths for generalized linear models via coordinate descent. *Journal of statistical software*, 33(1):1.

Ellen R Girden. 1992. *ANOVA: Repeated measures*. 84. Sage.

Song Han, Huizi Mao, and William J Dally. 2015. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*.

Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. 2017. Quantized neural networks: Training neural networks with low precision weights and activations. *JMLR*, 18(1):6869–6898.

Sachin Kamboj, Willett Kempton, and Keith S Decker. 2011. Deploying power grid-integrated electric vehicles as a multi-agent system. In *AAMAS*.

Yoon Kim, Yacine Jernite, David Sontag, and Alexander M Rush. 2015. Character-aware neural language models. *arXiv preprint arXiv:1508.06615*.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *arXiv preprint arXiv:1808.06226*.

Quoc V Le, Navdeep Jaitly, and Geoffrey E Hinton. 2015. A simple way to initialize recurrent networks of rectified linear units. *arXiv preprint arXiv:1504.00941*.

Jason Lee, Kyunghyun Cho, and Thomas Hofmann. 2017. Fully character-level neural machine translation without explicit segmentation. *Transactions of the Association for Computational Linguistics*, 5:365–378.

Gurvan L'Hostis, David Grangier, and Michael Auli. 2016. Vocabulary selection strategies for neural machine translation. *arXiv preprint arXiv:1610.00072*.

Wang Ling, Isabel Trancoso, Chris Dyer, and Alan W Black. 2015. Character-based neural machine translation. *arXiv preprint arXiv:1511.04586*.

Scott M Lundberg, Gabriel Erion, Hugh Chen, Alex DeGrave, Jordan M Prutkin, Bala Nair, Ronit Katz, Jonathan Himmelfarb, Nisha Bansal, and Su-In Lee. 2020. From local explanations to global understanding with explainable ai for trees. *Nature machine intelligence*, 2(1):2522–5839.

Scott M Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. In *NIPS*.

Moshe Mash, Yoram Bachrach, and Yair Zick. 2017. How to form winning coalitions in mixed human-computer settings. In *IJCAI*.

Tomomi Matsui and Yasuko Matsui. 2000. A survey of algorithms for calculating power indices of weighted majority games. *Journal of the Operations Research Society of Japan*, 43(1):71–86.

Yasuko Matsui and Tomomi Matsui. 2001. Np-completeness for calculating power indices of weighted majority games. *TCS*, 263(1-2):305–310.

Nicolai Meinshausen and Peter Bühlmann. 2010. Stability selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(4):417–473.

Dmitry Molchanov, Arsenii Ashukha, and Dmitry Vetrov. 2017. Variational dropout sparsifies deep neural networks. *arXiv preprint arXiv:1701.05369*.

Kirill Neklyudov, Dmitry Molchanov, Arsenii Ashukha, and Dmitry P Vetrov. 2017. Structured bayesian pruning via log-normal multiplicative noise. In *NIPS*.

Kim F Nimon and Frederick L Oswald. 2013. Understanding the results of multiple linear regression: Beyond standardized regression coefficients. *Organizational Research Methods*, 16(4):650–674.

Ivan Provilkov, Dmitrii Emelianenko, and Elena Voita. 2019. Bpe-dropout: Simple and effective subword regularization. *arXiv preprint arXiv:1910.13267*.

Juan Ramos et al. 2003. Using tf-idf to determine word relevance in document queries. In *Instructional conference on machine learning*, pages 133–142.

Wojciech Samek, Grégoire Montavon, Andrea Vedaldi, Lars Kai Hansen, and Klaus-Robert Müller. 2019. *Explainable AI: interpreting, explaining and visualizing deep learning*, volume 11700. Springer Nature.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*.

Lloyd S Shapley. 1953. A value for n-person games. *Contributions to the Theory of Games*, 2(28):307–317.

Xing Shi and Kevin Knight. 2017. Speeding up neural machine translation decoding by shrinking run-time vocabulary. In *ACL*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *EMNLP*.

Philip D Strafiin Jr. 1988. The shapley—shubik and banzhaf power indices as probabilities. *The Shapley value: essays in honor of Lloyd S. Shapley*, page 71.

Andros Tjandra, Sakriani Sakti, and Satoshi Nakamura. 2017. Compressing recurrent neural network with tensor train. In *IJCNN*.

Suguru Ueda, Makoto Kitaki, Atsushi Iwasaki, and Makoto Yokoo. 2011. Concise characteristic function representations in coalitional games based on agent types. In *IJCAI*, volume 11, pages 393–399.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. 2019. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641.

E Roy Weintraub. 1992. *Toward a history of game theory*, volume 24. Duke University Press.

Wei Wen, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. 2016. Learning structured sparsity in deep neural networks. *arXiv preprint arXiv:1608.03665*.

Dani Yogatama, Manaal Faruqui, Chris Dyer, and Noah Smith. 2015. Learning word representations with hierarchical sparse coding. In *ICML*.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *NIPS*.