Les Cahiers du GERAD ISSN: 0711–2440

Optimal text-based time-series indices

D. Ardia, K. Bluteau

G–2024–32

Mai 2024

GERAD HEC Montréal
3000, chemin de la Côte-Sainte-Catherine
Montréal (Québec) Canada H3T 2A7

Tél. : 514 340-6053
Téléc. : 514 340-5665
info@gerad.ca
www.gerad.ca

Optimal text-based time-series indices

David Ardia [a]

Keven Bluteau [b]

[a] *GERAD & Department of Decision Sciences, HEC Montréal, Montréal (Qc), Canada, H3T 2A7*

[b] *Department of Finance, Université de Sherbrooke, Sherbrooke (QC), Canada, J1K 2R1*

david.ardia@hec.ca
keven.bluteau@usherbrooke.ca

Mai 2024
Les Cahiers du GERAD
G–2024–32

Abstract : We propose an approach to construct text-based time-series indices in an optimal way—typically, indices that maximize the contemporaneous relation or the predictive performance with respect to a target variable, such as inflation. We illustrate our methodology with a corpus of news articles from the Wall Street Journal by optimizing text-based indices focusing on tracking the VIX index and inflation expectations. Our results highlight the superior performance of our approach compared to existing indices.

Acknowledgements: We are grateful to IVADO and the Natural Sciences and Engineering Research Council of Canada (grant RGPIN–2022–03767) for their financial support. Please contact the corresponding author to have access to the computer code and the data sets.

1 Introduction

In economic and financial research, there is a growing trend of integrating textual data such as news articles into econometrics analysis (see Gentzkow, Kelly, and Taddy, 2019, for a review). This integration is typically done by (i) selecting, (ii) transforming, and (iii) aggregating textual content into a time-series representation (see Ardia, Bluteau, and Boudt, 2019; Algaba et al., 2020, for a general overview of these steps). While many studies have focused on steps (ii) and (iii)—transforming and aggregating textual data into a quantitative measure such as sentiment (see e.g., Loughran and McDonald, 2014; Jegadeesh and Wu, 2013; Manela and Moreira, 2017)—the essential selection step (i), which usually relies on subjective ad-hoc rules, has not received much attention yet.

We aim to fill this gap in this article by proposing an approach to construct text-based time-series indices optimally. Specifically, our algorithm determines which set of texts, among a large corpus, leads to a text-based index that is optimal for a specific objective—typically, an index that maximizes the contemporaneous relation or the predictive performance with respect to a target variable, such as inflation. Our methodology relies on binary *selection matrices* that, applied to the vocabulary of tokens, select the relevant texts in the corpus. Various widely-known text-based indices, such as the Economic Policy Uncertainty (EPU) index by Baker, Bloom, and Davis (2016), can be formulated in terms of selection matrices.

Optimizing selection matrices is challenging due to the inherent non-linearity that arises when aggregating selected texts into text-based indices. To overcome this difficulty, we design a genetic algorithm with domain-specific knowledge to explore the solution space and obtain the *optimal* selection matrix. The algorithm starts with an initial population of selection matrices. These matrices are evaluated using a fitness function that measures the resulting textual-index performance in achieving the objective of selecting the texts. At each iteration, the population of selection matrices undergoes tailor-made crossover and mutation operations, as traditional operators are not well suited to this optimization problem. We also implement additional steps to address potential overfitting issues and leverage the information in word embeddings of promising solutions to help explore good solutions more efficiently. Finally, we introduce a pruning step to avoid sub-optimal solutions.

To showcase the relevance of our methodology, we conduct two empirical applications using a collection of 763,542 Wall Street Journal news articles spanning from January 2000 to August 2021.

First, we validate our methodology with the EPU index. Specifically, we use our algorithm to see if we can recover the set of keywords proposed by Baker, Bloom, and Davis (2016) when building their EPU index. Given the number of tokens in our corpus vocabulary, recovering the set of tokens for the three dictionary dimensions of Baker, Bloom, and Davis (2016) is not trivial. We show that our approach (i) can recover the set of keywords proposed by the authors—their selection matrix—and (ii) does so in a reasonable computational time.

Second, we evaluate the performance of our methodology against established benchmarks, focusing on the monthly VIX index and inflation expectations derived from the Michigan Surveys of Consumers. In this comparative analysis, the text-based indices generated by our methodology consistently outperform the text-based benchmarks on the test window, a time period not considered during the optimization of the selection matrices.

The rest of this paper is organized as follows. Section 2 introduces the notation, presents the concept of selection matrices, and outlines the optimization problem. Section 3 presents our optimization strategy, including the genetic algorithm as well as the proposed crossover and mutation operators. Section 4 presents our empirical applications, and Section 5 concludes.

2 Token-based text selection

We first introduce the concept of tokens and vocabulary to analyze a corpus of texts. A token, denoted by v, represents a sequence of characters, including acronyms, words, sequences of words, or even regular expressions. The vocabulary of size V is defined as a collection of such tokens.

The text corpus is represented as a matrix $\boldsymbol{C}_t$ of size $N_t \times V$, where N_t is the number of texts available at a given time t. Each element $c_{n,v,t}$ among a collection of matrices $\boldsymbol{C}_t$, where $t = 1, \ldots, T$, indicates if the token v appears in the text n published at time t. If it does, the element takes the value of one and zero otherwise. Hence, a specific text published at time t can be represented by the row vector $\boldsymbol{c}_{n,t}$ (of size $1 \times V$).

Typically, the corpus consists of a vast collection of texts, and our objective is to select texts for further analysis. For instance, we could focus on texts related to the U.S. economy from a collection of news articles published by various newspapers. A simple way to proceed is by using a keyword-based (i.e., token-based in our nomenclature) approach, which we formalize below.

We employ a selection matrix $\boldsymbol{\Omega}$ of size $V \times K$, where each column vector $\boldsymbol{\omega}^k$ (of size $V \times 1$) corresponds to a selection rule. The binary selection vector $\boldsymbol{\kappa}_t$ of size N_t contains elements $\kappa_{n,t}$, which are defined through the selection function $f_\kappa(\cdot)$ as:

$$\kappa_{n,t} \equiv f_\kappa(\boldsymbol{\Omega}, \boldsymbol{c}_{n,t}) \equiv I\left[\sum_{k=1}^{K} I\left[\boldsymbol{c}_{n,t}\boldsymbol{\omega}^k \geq 0\right] = \sum_{k=1}^{K} I\left[\sum_{v=1}^{V} \boldsymbol{i}_V \boldsymbol{\omega}^k > 0\right]\right], \tag{1}$$

where $I[\cdot]$ is an indicator function that takes a value of one if the condition inside the parenthese is true and zero otherwise.

In (1), $\sum_{k=1}^{K} I\left[\boldsymbol{c}_{n,t}\boldsymbol{\omega}^k \geq 0\right]$, counts the number of unique active tokens (i.e., non-zero value) for dimension k of the selection matrix $\boldsymbol{\Omega}$ (represented by $\boldsymbol{\omega}^k$) that appear in the text n at time t (represented by row vector $\boldsymbol{c}_{n,t}$). If this count is greater than zero for all dimensions $k = 1, \ldots, K$, each element of the sum takes the value of one, and the summation takes the value K. The second part, $\sum_{k=1}^{K} I\left[\sum_{v=1}^{V} \boldsymbol{i}_V \boldsymbol{\omega}^k > 0\right]$, where $\boldsymbol{i}_V$ is a row vector of lenght V where each element is equal to one, counts the number of active tokens within each dimension of the selection matrix $\boldsymbol{\Omega}$. If this count is greater than zero for all dimensions, each element of the sum takes value one, and the sum is K. Thus, if the text n published at time t contains at least one active token within each non-zero column k of $\boldsymbol{\Omega}$, the selection condition is satisfied, and $\kappa_{n,t}$ is equal to one.

The well-known Economic Policy Uncertainty (EPU) index of Baker, Bloom, and Davis (2016) relies on this selection function. The EPU selection condition is defined by a vocabulary of size 10 with three selection dimensions ($V = 10$ and $K = 3$) representing economic, policy, and uncertainty-related tokens. Using our notation, the selection matrix $\boldsymbol{\Omega}$ (of size 10×3) of the EPU index is:

$$\boldsymbol{\Omega}_{\text{EPU}} \equiv \begin{array}{c} \\ \text{economic} \\ \text{economy} \\ \text{congress} \\ \text{deficit} \\ \text{federal\_reserve} \\ \text{legislation} \\ \text{regulation} \\ \text{white\_house} \\ \text{uncertain} \\ \text{uncertainty} \end{array} \begin{array}{c} \begin{array}{ccc} \text{Economy} & \text{Policy} & \text{Uncertainty} \end{array} \\ \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix} \end{array}.$$

The EPU selection matrix contains 2, 6, and 2 active tokens in the first, second, and third dimension, respectively. In that case, the second terms of (1): $\sum_{k=1}^{K} I\left[\sum_{v=1}^{V} \boldsymbol{i}_V \boldsymbol{\omega}_{\text{EPU}}^k > 0\right] = 3$. For a text n

published at time t containing the tokens "economic," "congress," and "uncertainty," the first terms of (1): $\sum_{k=1}^{K} I\left[\boldsymbol{c}_{n,t}\boldsymbol{\omega}_{\text{EPU}}^{k} \geq 0\right] = 3$. Given the equality of both terms, for this article, $\kappa_{n,t} = 1$. Alternatively, a text n published at time t containing two times the token "economic" and one time the token "congress," the first terms of (1), $\sum_{k=1}^{K} I\left[\boldsymbol{c}_{n,t}\boldsymbol{\omega}_{\text{EPU}}^{k} \geq 0\right] = 2$, and thus $\kappa_{n,t} = 0$ as both the left and right terms of (1) are not equal.

Several news-media-based indices follow similar selection rules: the Climate Policy Uncertainty index of Gavriilidis (2021), the Equity Market Volatility index of Baker et al. (2019), the Monetary Policy Uncertainty index of Husted, Rogers, and Sun (2020), and the Trade Policy Uncertainty index of Handley and Limão (2022).

The selection function in (1) does not, however, take into account more complex selection mechanisms, such as the addition of a proximity condition between the tokens of two or more dimensions of the selection matrix $\boldsymbol{\Omega}$, as the one used to construct the Geopolitical Risk index of Caldara and Iacoviello (2022). For instance, in the case of the EPU index, one might require at least one token from the "Economy" dimension detected in the neighborhood (e.g., within a two-token distance) of one of the "Policy" tokens in the same news article. Such additional complexity in the selection rule could be integrated using the information from a token distance matrix similar to the one used in Ardia, Bluteau, and Kassem (2021). To keep the exposition simple, we do not consider this type of variation of the selection function $f_{\kappa}(\cdot)$ in this paper.

2.1 Downstream transformation

After identifying relevant texts in the corpus with the selection function $f_{\kappa}(\cdot)$, we typically transform and aggregate the selected texts into a quantitative time-series measure. We present below attention-based and context-based time-series measures.

An attention measure aims to measure the level of importance attributed to the selected texts across all texts in the corpus at a given time. For instance, it could be to measure the level of importance given to the topic of climate change by the news media (see, e.g., Ardia et al., 2023). Given our prior notation and definition, a typically used measure of attention is:

$$f_{\text{att}}(\boldsymbol{\Omega}, \boldsymbol{C}_t) \equiv \frac{1}{N_t} \sum_{n=1}^{N_t} f_{\kappa}(\boldsymbol{\Omega}, \boldsymbol{c}_{n,t}) \,. \tag{2}$$

At each time t, we measure how many texts are selected and normalize this number by the total number of texts in the corpus at that time. If the corpus is composed of several sources (e.g., various newspapers), it may be relevant to standardize each source before the aggregation in (2), as in Baker, Bloom, and Davis (2016).

An alternative form of transformation is the content transformation, which requires more information than the output of the selection step. In particular, the content of selected texts is processed to derive a "score," such as polarity or sentiment (see, e.g. Algaba et al., 2020). The content transformation function can be written as:

$$f_{\text{con}}(\boldsymbol{\Omega}, \boldsymbol{C}_t, \boldsymbol{\zeta}) \equiv \frac{1}{\sum_{n=1}^{N_t} f_{\kappa}(\boldsymbol{\Omega}, \boldsymbol{c}_{n,t})} \sum_{n=1}^{N_t} \left(f_{\kappa}(\boldsymbol{\Omega}, \boldsymbol{c}_{n,t}) \sum_{v=1}^{V} c_{n,v,t}\, \zeta_v \right) , \tag{3}$$

where ζ_v are token weights (integer or real numbers) used to score the selected texts. The average score of selected texts at time t is then used to obtain an index. Token weights can be measured from manually-composed lexicons (e.g., Loughran and McDonald, 2014) or in a data-driven way (e.g., Jegadeesh and Wu, 2013; Manela and Moreira, 2017; Kelly, Manela, and Moreira, 2021; Ardia, Bluteau, and Boudt, 2022). Much of the literature has focused on estimating $\boldsymbol{\zeta} \equiv (\zeta_1, \ldots, \zeta_V)$ given a predefined selection of texts.

A practical advantage of employing the attention transformation over the content transformation lies in its reduced reliance on processing the entire corpus of texts through an additional layer of models to derive $\boldsymbol{\zeta}$ (see, e.g., Algaba et al., 2020). For instance, when utilizing a news articles aggregator that consolidates the desired news sources, the data required for computing attention solely invoke queries derived from a selection matrix (to determine the count of selected articles) and statistical information on the number of published news articles from the desired sources. This streamlined approach facilitates the timely computation and updating of the attention index. See, for instance, https://www.policyuncertainty.com/, where multiple indices similar to the EPU are updated monthly.

When using a data-driven method for content transformation, the joint estimation of the selection matrix $\boldsymbol{\Omega}$, and content transformation weights $\boldsymbol{\zeta}$, is preferable, albeit computationally intensive. Because of the added complexity of the content transformation, which is a challenge in this study, to isolate the effect of news selection from content transformation, we focus on the attention transformation measure.

2.2 Objective-based tokens optimization

The objective of deriving information in texts by a selection function and a transformation process is to capture a (contemporaneous or predictive) relation between that information and a variable of interest y_t $(t = 1, \ldots, T)$. A selection matrix is typically based on a subjective but guided assessment of the tokens necessary to capture the information we want to extract from the texts. Formally, given a relevant transformation function $f(\boldsymbol{\Omega}, \boldsymbol{C}_t)$, the aim is to minimize:

$$\min_{\alpha,\beta} \frac{1}{T} \sum_{t=1}^{T} \left(y_t - \left(\alpha + \beta f(\boldsymbol{\Omega}, \boldsymbol{C}_{t-h})\right)\right)^2 , \tag{4}$$

with $h \geq 0$, which is a simple linear regression problem.[1] Once estimated, one would typically make inference about parameter β or test whether the (out-of-sample) root-mean-squared error is significantly lower than the root-mean-squared error of a model where β is constrained to be zero (i.e., nested model); see Ardia, Bluteau, and Boudt (2019). For instance, several studies analyze the relationship between the Economic Policy Uncertainty and economic variables using the following regression framework:[2]

$$\min_{\alpha,\beta} \frac{1}{T} \sum_{t=1}^{T} \left(y_t - \left(\alpha + \beta f_{\text{att}}(\boldsymbol{\Omega}_{\text{EPU}}, \boldsymbol{C}_{t-h})\right)\right)^2 . \tag{5}$$

One limitation of this approach is related to the fact that $\boldsymbol{\Omega}_{\text{EPU}}$ is given. This may be reasonable in some applications but suboptimal for others, particularly when it comes to nowcasting and forecasting. We thus introduce the following optimization problem:

$$\min_{\alpha,\beta,\boldsymbol{\Omega}} \frac{1}{T} \sum_{t=1}^{T} \left(y_t - \left(\alpha + \beta f_{\text{att}}(\boldsymbol{\Omega}, \boldsymbol{C}_{t-h})\right)\right)^2 + \lambda_1 \sum_{v=1}^{V} I[\boldsymbol{\omega}_v \boldsymbol{i}_K' > 1] + \lambda_2 I[\beta > 0] + \lambda_3 I[\beta \leq 0] . \tag{6}$$

where $\boldsymbol{\omega}_v$ denotes the vth row of $\boldsymbol{\Omega}$. In optimization problem (6), we assume an unrestrictive and large vocabulary and optimize active tokens selection in addition to the regression parameters. We use three penalty terms to control the optimization process. We standardize y_t to avoid scale-dependent penalty parameters.

[1] While we present here the case of a linear relationship between y_t and $f(\boldsymbol{\Omega}, \boldsymbol{C}_{t-h})$, more general specifications would work just as well, including multiple linear regression model by adding additional explanatory variables or by not assuming a linear relationship. To ease the presentation, we focus on a simple linear relationship without additional explanatory variables.

[2] At the time of this writing, for instance, Baker, Bloom, and Davis (2016), the research paper introducing the EPU index, has received over 9,000 citations according to Google Scholar. While many of those citations do not use the index per se, many, however, analyze how the EPU explains or interacts with other macroeconomics and financial variables.

First, we introduce a penalty term $\lambda_1 \sum_{v=1}^{V} I[\boldsymbol{\omega}_v \boldsymbol{i}_K' > 1]$, which aims to avoid the activation of the same token across multiple dimensions in the selection matrix. A higher value of λ_1 enforces a constraint that restricts each dimension from overlapping. This constraint can be beneficial when assigning specific topics to each dimension. However, it also limits the flexibility of the selection matrix. Consider the following two three-dimensional selection matrices:

$$\begin{array}{c} \\ \text{economic} \\ \text{stock\_market} \\ \text{federal\_reserve} \\ \text{crisis} \end{array} \begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \begin{pmatrix} 1 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \end{array} \qquad \begin{array}{c} \\ \text{economic} \\ \text{stock\_market} \\ \text{federal\_reserve} \\ \text{crisis} \end{array} \begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \end{array}$$

With the first selection matrix, a text containing the token "economic" would be selected, while it would not necessary with the second, as in this case, a text needs to include tokens from the other dimensions, distinct from "economic." Therefore, a lower value of λ_1 allows for the inclusion of selection conditions with lower dimensions than the one of the selection matrix, meaning, for instance, a text may fulfill the requirement with a single distinct token instead of three distinct tokens (when $K = 3$). Setting $\lambda_1 = 0$ provides an opportunity for a higher-dimensional selection matrix to mimic a lower-dimensional one, as shown by these two equivalent solutions:

$$\boldsymbol{\Omega} = \begin{array}{c} \\ \text{economic} \end{array} \begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \begin{pmatrix} 1 & 1 & 1 \end{pmatrix} \end{array} = \begin{array}{c} \\ \text{economic} \end{array} \begin{array}{c} \begin{array}{c} 1 \end{array} \\ \begin{pmatrix} 1 \end{pmatrix} \end{array}$$

Increasing flexibility with $\lambda_1 = 0$ increases complexity, which may lead to challenges in the selection matrix's out-of-sample performance. A higher value of λ_1 penalizes the score of the left selection matrix and, therefore, leads to a narrower set of potential solutions.

The two remaining penalty terms, $\lambda_2 I[\beta > 0]$ and $\lambda_3 I[\beta \leq 0]$, serve to establish the intended association between the text-based index and the variable of interest y_t. When a negative relationship is sought, the condition $\lambda_2 > \lambda_3$ is applied. Consequently, this allows for controlling a selection condition that selects texts either positively or negatively correlated with the variable of interest. It is crucial to choose the sign of beta as selection matrices can be derived in a manner that selects texts associated negatively with the variable of interest and another set that is positively associated with it.

3 Optimization strategy

Regression (4) is trivial to estimate as $\boldsymbol{\Omega}$ is fixed and thus not part of the estimation process. On the contrary, the minimization problem (6) is complex as $f(\boldsymbol{\Omega}, \boldsymbol{C}_t)$ is a non-linear function of $\boldsymbol{\Omega}$, which is now considered as a parameter. Below, we present our strategy based on a genetic algorithm to perform the minimization. We first define the crossover and mutation operators specifically designed for our problem.

3.1 Crossover and mutation operators

Because of the distinctive characteristics of our optimization problem, we introduce specific crossover and mutation operators. To illustrate these operators, we consider a specific scenario wherein our optimization objective is to replicate the EPU index while imposing a constraint that restricts the number of active tokens to five out of a maximum of VK (i.e., when all elements in the matrix $\boldsymbol{\Omega}$ are equal to one).

3.1.1 Token crossover operator

We begin with the "token crossover operator." This operator takes two parent solutions and generates two offspring solutions by exchanging a single token between the parents while maintaining the

dimensionality of the replaced token. Consider the following two parents:

Parent 1:	1	2	3
economic	1	0	0
legislation	0	1	0
congress	0	1	0
house	0	0	1
risk	0	0	1
⋮	⋮	⋮	⋮

Parent 2:	1	2	3
economy	1	0	0
financial_crisis	0	1	0
congress	0	1	0
house	0	0	1
risk	0	0	1
⋮	⋮	⋮	⋮

The token crossover operator selects two tokens (in gray) and generates two offspring solutions as follows:

Child 1	1	2	3
financial_crisis	1	0	0
legislation	0	1	0
congress	0	1	0
house	0	0	1
risk	0	0	1
⋮	⋮	⋮	⋮

Child 2	1	2	3
economy	1	0	0
economic	0	1	0
congress	0	1	0
house	0	0	1
risk	0	0	1
⋮	⋮	⋮	⋮

It is important to note that this crossover operator maintains a fixed number of active tokens within each dimension and operates solely on already activated tokens. In contrast, a regular crossover operator would not guarantee that each dimension remains the same size or even has active tokens. Additionally, owing to the high sparsity of our optimization problem (where only five tokens are active out of a potential VK points), a regular crossover operator could perform non-altering operations by exchanging slices of non-active tokens (i.e., tokens with zero values), consequently reducing the efficiency of the search algorithm.

3.1.2 Mutation operators

Next, we focus on three mutation operators, each serving a distinct purpose.

Switch mutation operator The "switch mutation" operator allows an active token to change its dimension:

Parent	1	2	3
economy	1	0	0
economic	0	1	0
congress	0	1	0
house	0	0	1
risk	0	0	1
⋮	⋮	⋮	⋮

Child	1	2	3
economy	1	0	0
economic	1	0	0
congress	0	1	0
house	0	0	1
risk	0	0	1
⋮	⋮	⋮	⋮

This operator allows for the correction of potential dimensional misalignment. It is worth noting that in the example above, the set of texts selected by the parent, while not optimal, may be highly correlated with the set of texts of the child, which is optimal. This operation facilitates a more efficient search for misalignment compared to a random search. Similar to other operators, this mutation operator only applies to active tokens. However, it cannot change the dimensions of a token if it is the only token within its dimension, ensuring that each dimension has at least one active token.

N-gram mutation operator Next, we introduce the "n-gram mutation" operator, which involves transforming one of the selected tokens into an n-gram containing that token:

$$\begin{array}{c} \text{Parent} \\ \text{economy} \\ \text{economic} \\ \text{congress} \\ \text{house} \\ \text{risk} \\ \vdots \end{array} \begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ \vdots & \vdots & \vdots \end{pmatrix} \end{array} \qquad \begin{array}{c} \text{Child} \\ \text{economy} \\ \text{economic} \\ \text{congress} \\ \text{white\_house} \\ \text{risk} \\ \vdots \end{array} \begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ \vdots & \vdots & \vdots \end{pmatrix} \end{array}$$

To comprehend this operator, we must recognize that the texts containing an n-gram always include the texts containing its components. For instance, the number of texts containing the token "stock" is equal to or larger than the number of texts containing the bi-gram "stock_market." As such, if the algorithm selects the component of an n-gram, there is a high likelihood that using an n-gram containing that component would be more optimal. The n-gram mutation operator increases the possibility of testing an n-gram compared to testing any other token. Similar to the crossover operator, the n-gram mutation only applies to active tokens, but it also applies solely to tokens that are part of n-grams within the vocabulary.

Transform mutation operator Finally, we have the "transform mutation" operator, which changes a token from a specific dimension to another token in that same dimension:

$$\begin{array}{c} \text{Parent} \\ \text{economy} \\ \text{economic} \\ \text{congress} \\ \text{white\_house} \\ \text{risk} \\ \vdots \end{array} \begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ \vdots & \vdots & \vdots \end{pmatrix} \end{array} \qquad \begin{array}{c} \text{Child} \\ \text{economy} \\ \text{economic} \\ \text{congress} \\ \text{white\_house} \\ \text{uncertainty} \\ \vdots \end{array} \begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ \vdots & \vdots & \vdots \end{pmatrix} \end{array}$$

The transform mutation is the only operator that activates new tokens (not n-grams) that have not already been activated within the population. This operator thus introduces new tokens within the population by replacing an active token from a candidate selection matrix with a new one. The transform mutation ensures that the number of active tokens within each dimension remains the same. This operator is crucial in controlling the number of active tokens while allowing the introduction of new active tokens. A standard mutation operator that flips one bit would not work in this case. Consider a vocabulary of size 4,800 with three dimensions, leading to $4{,}800 \times 3 = 14{,}400$ potential number of active points. If a parent has ten active tokens out of 14,400, the standard mutation operator has a $\frac{14{,}400-10}{14{,}400}) = 99.93\%$ chance of generating a child with 11 active tokens and a 0.07% chance of generating a child with nine active tokens. Over a large number of iterations, this leads to a search space with a large number of active tokens, ultimately leading to overfitting.

By design, many of these operators only apply to tokens that are already active within the corpus to ensure an efficient search. Due to the extreme sparsity of our optimization problem, without these operators, the likelihood of performing non-altering operations is extremely high. We also note that these operations ensure that the number of active tokens in the child will always be the same as the number of active tokens in the parent. This is crucial for our search strategy, which is presented below.

3.2 Calibration

The calibration strategy is centered around (i) efficient search and (ii) avoiding overfitting. First, we define a training and a validation window. We start the optimization process by defining the initial

population of $q = 1, \ldots, Q$ candidate selection matrices $\boldsymbol{\Omega}_q$ of size $V \times K$. Each member of the initial population starts with K active tokens (non-zero value in the selection matrix), where K is the number of dimensions of the selection matrix. Each dimension is set to have one active token. For instance, for $K = 3$, these could be members of the initial population (displaying only active tokens):

$$\begin{array}{c} \\ \text{democrats} \\ \text{european\_union} \\ \text{policy} \\ \vdots \end{array} \begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ \vdots & \vdots & \vdots \end{pmatrix} \end{array} \qquad \begin{array}{c} \\ \text{religion} \\ \text{market} \\ \text{presidency} \\ \vdots \end{array} \begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ \vdots & \vdots & \vdots \end{pmatrix} \end{array}$$

while these would not:

$$\begin{array}{c} \\ \text{democrats} \\ \text{european\_union} \\ \text{policy} \\ \vdots \end{array} \begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ \vdots & \vdots & \vdots \end{pmatrix} \end{array} \qquad \begin{array}{c} \\ \text{religion} \\ \text{market} \\ \text{presidency} \\ \text{coffee} \\ \vdots \end{array} \begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ \vdots & \vdots & \vdots \end{pmatrix} \end{array}$$

Inpired from Scrucca (2013, 2017), we perform genetic optimization as follows:

1. Begin an epoch with an initial population. For the first epoch, this is randomly initialized, while for subsequent epochs, the initial population depends on the results from the previous epoch. Each epoch contains H iterations along these steps:
   (a) For each candidate $q = 1, \ldots, Q$ in the population, compute $f(\boldsymbol{\Omega}_q, \boldsymbol{C}_{t-h})$.
   (b) For each candidate $q = 1, \ldots, Q$ in the population, estimate parameters α and β by ordinary least squares on the training data with the computed $f(\boldsymbol{\Omega}_{\boldsymbol{q}}, \boldsymbol{C}_{t-h})$ as input.
   (c) For each candidate $q = 1, \ldots, Q$ in the population, compute the objective function (6) on the training and validation sets using the estimated parameters from step 2.
   (d) Elitism step. Store the 5% lowest objective value chromosome in the training window as well as the best solution. For the best solution, store its validation window objective value. 5% is a common choice for elitism steps in genetic algorithms.
   (e) Tournament selection. Select at random three members of the population, then select the one with the best objective function. Do this Q times to generate an interim population.
   (f) Token crossover: For pairs of two randomly chosen selection matrices in the interim population, perform a token crossover operation with probability S to update the interim population, where $S = \frac{Q_{\text{unique}}}{Q}$ is the ratio of the unique selection matrix in the interim population over the population size. Since doing crossover on two populations that are exacty the same does not do anything, the probability of doing a crossover operation scale. If all matrices are unique ($S = 1$), we would perform crossover on all population members (100% probability); if all matrices are the same, $S \approx 0$, no crossover is performed.
   (g) Switch mutation. For each selection matrix in the interim population, perform a switch mutation operation with 5% probability to update the interim population. 5% is chosen arbitrarily.
   (h) N-grams mutation. For each selection matrix in the interim population, perform n-grams mutation operation with 5% probability to update the interim population. 5% is chosen arbitrarily.

(i) Transform mutation. For each selection matrix in the interim population, perform a transform mutation with $1 - S\%$ probability to update the interim population. The transform mutation changes an active word for another non-active word. If all matrices are the same (S ≈ 0), all members of the interim population will have a transform mutation operator applied to them. If all matrices are unique, no transform mutation will be applied. This overall strikes a balance between the crossover and transform mutation operator, thus leveraging crossover when appropriate and doing the equivalent of pure random search when there is no value to performing the crossover operator.

(j) Elitism step. Replace 5% instance of the interim population with the 5% lowest objective value solution stored in step (d). We keep the best members of the populations before crossover and mutation. This strategy usually speeds up the convergence of the algorithm.

(k) Start a new iteration with the new population until we reach the last iteration.

2. From H best solutions, each one stored at step (d), Select the one with the lowest validation window objective value.
3. Refine the vocabulary for the next epoch. To achieve this, we employ word embeddings to reduce the vocabulary size while retaining information from the active tokens of the best solution in the preceding step. A word embedding space maps tokens into high-dimensional vectors, where similar tokens exhibit shorter distances between their vectors. First, we compute the average token vector for each dimension of the best solution in step 2. Then, for each dimension, we calculate the cosine similarity between the average token embedding of that dimension and all the tokens in the original vocabulary of size V. We retain tokens with a cosine similarity larger than 0.2. The set of unique, similar tokens across all dimensions, constitutes our new vocabulary. Effectively, this step narrows down the search space to a more relevant vocabulary, considering that the optimally selected tokens are indicative of the relevant tokens concerning the dependent variable.
4. Subsequently, we form a new population of size Q using the new vocabulary, where each member starts with $K + 1$ active tokens, one more than the old population. Additionally, we insert the optimal solution from step 2 within the new population. If we find a better solution with more active tokens, it will disappear from the population.
5. Start a new epoch until we achieve the maximum number of desired active tokens (from K up to 15 in our application).

The solution with the lowest objective value from the validation window from step 2 of the last epoch is considered as the interim optimal solution. To obtain the final solution, we perform a pruning step. This step aims to eliminate tokens that may have adversely affected the model fit but were added due to the random nature of the optimization process (e.g., uninformative tokens added before informative ones). To accomplish this, we test variations of the optimal selection matrix by deactivating one or several active tokens, encompassing all variations involving only deactivations. For instance, if the solution with the lowest objective value from the validation window has 12 active tokens, we test $2^{12} = 4{,}096$ potential new solutions (ranging from all tokens being active to all tokens being deactivated).

4 Empirical applications

In this section, we present two empirical applications of our methodology. First, we validate our approach with the EPU index. Second, we apply our optimization process to (i) tracking the VIX and (ii) nowcasting inflation expectations. We use the latter two usecases as news-based indices have been used in previous studies in relation to those variables, and as such, we can test our method against benchmarks. Beforehand, we describe the corpus of text used throughout this section and the data processing steps.

4.1 Textual data, corpus processing, and vocabulary

Our corpus consists of all news articles published in the printed version of the Wall Street Journal from January 2000 to August 2021. In total, the corpus is composed of 763,542 news articles. To derive candidate tokens for the vocabulary describing the corpus, we proceed as in Ardia, Bluteau, and Meghani (202x):

1. To normalize the news articles, we lowercase and lemmatize each word into its root form.
2. We then detect and combine collocation in our corpus. For instance, the compound words "interest rate" or "exchange rate" provide more information about the content of an article than if these words were not considered a combination. We rely on two methods to detect collocations in our documents: (i) The RAKE (Rapid Automatic Keyword Extraction) algorithm (Rose et al., 2010) and (ii) the process described in Hansen, McMahon, and Prat (2018, Section IV.B).[3] For each method, we select the collocations that occur at least 100 times for bigrams and 50 times for trigrams. Finally, we concatenate the individual tokens of the collocation in each text of our corpus (e.g., "interest rate" becomes "interest_rate").
3. We then build a matrix representation of the text where each column is a token, and each row is a news article such that each element represents the number of times a token is observed for a corresponding news article. We eliminate tokens that appear in less (more) than 1% (20%) of the news articles.
4. Finally, we manually verify each token and remove non-informative tokens such as dates or tokens with numbers. We also remove any token related to specific events, persons, locations, companies, or products. Overall, this results in a vocabulary of size $V = 2{,}261$.

We use the pre-trained continuous bag-of-word embedding model of Rahimikia, Zohren, and Poon (2021) available at https://fintext.ai to retrieve the token vector for each of the 2,261 tokens. The model is compiled from 15 years of business news archives and, as a domain-specific model, is more appropriate to capture the relationship between tokens than a pre-trained general-domain word embedding model such as Google Word2Vec. For collocations, we take the average of the token vector across the individual collocation tokens. These token vector representations of the 2,261 tokens will be used to narrow down the vocabulary after each epoch.

4.2 Validation of our algorithm With the economic policy uncertainty

Our first application aims to validate our approach by determining if we can recover the EPU selection matrix. Specifically, we will perform the following optimization problem:

$$
\begin{aligned}
(\hat{\alpha}, \hat{\beta}, \hat{\boldsymbol{\Omega}}) \equiv \underset{\alpha,\beta,\boldsymbol{\Omega}}{\arg\min} \frac{1}{T} \sum_{t=1}^{T} \left(f_{\text{att}}(\boldsymbol{\Omega}_{\text{EPU}}, \boldsymbol{C}_t) - \left(\alpha + \beta f_{\text{att}}(\boldsymbol{\Omega}, \boldsymbol{C}_t)\right)\right)^2 \\
+ \lambda_1 \sum_{v=1}^{V} I[\boldsymbol{i}_K \boldsymbol{\omega}_v > 1] + \lambda_2 I[\beta > 0] + \lambda_3 I[\beta \leq 0] \,,
\end{aligned}
\tag{7}
$$

and see if $\hat{\boldsymbol{\Omega}} = \boldsymbol{\Omega}_{\text{EPU}}$. With $V = 2{,}261$ and $K = 3$, the number of selection matrices is $2^{2261 \times 3}$. Convergence toward the true EPU selection matrix would thus indicate massive improvement compared to a naive random search.

The setup for this experiment is as follows. We set the size of the selection matrices population to $Q = 300$, the number of iterations per epoch to $H = 200$, and the number of epochs to 15. The training

[3] Specifically, RAKE assigns a score to each candidate keyword based on the frequency of occurrence of the keyword in the text, as well as the frequency of occurrence of each of its constituent words in the text. The algorithm then identifies sets of adjacent keywords with high scores, indicating that they are likely collocations. The approach in Hansen, McMahon, and Prat (2018, Section IV.B) looks at part of speech patterns within the text. Following Justeson and Katz (1995) these patterns are: adjective-noun, noun-noun, adjective-adjective-noun, adjective-noun-noun, noun-adjective-noun, noun-noun-noun, and noun-preposition-noun. We find the part of speech of each word using the UDPipe methodology implemented in the R package **udpipe** (Wijffels, 2021).

window is composed of monthly data from January 2000 to December 2010 (132 observations), and the validation window ranges from January 2011 to December 2014 (48 observations). The remaining data from January 2015 to August 2021 is our test window (80 observations). We set $\lambda_1 = 0.25$, $\lambda_2 = 0$, and $\lambda_3 = 1$, so that we force the algorithm to search for tokens that have a positive relationship to the dependent variable. We test our algorithm for selection matrices with dimensions $K = 1, 2, 3$.

The optimal selection matrices obtained for each dimension are shown below:

$$\hat{\boldsymbol{\Omega}}_{K=1} = \begin{array}{c} \\ \text{uncertainty} \\ \text{uncertain} \\ \text{loom} \\ \text{unknown} \\ \vdots \end{array} \begin{array}{c} \begin{array}{c} 1 \end{array} \\ \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ \vdots \end{pmatrix} \end{array}$$

$$\hat{\boldsymbol{\Omega}}_{K=2} = \begin{array}{c} \\ \text{uncertainty} \\ \text{uncertain} \\ \text{monetary} \\ \text{banking} \\ \text{regulate} \\ \text{economic\_growth} \\ \text{spending} \\ \text{overhaul} \\ \text{extend} \\ \text{necessary} \\ \text{reserve} \\ \text{democrats} \\ \vdots \end{array} \begin{array}{c} \begin{array}{cc} 1 & 2 \end{array} \\ \begin{pmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \\ \vdots & \vdots \end{pmatrix} \end{array}$$

$$\hat{\boldsymbol{\Omega}}_{K=3} = \begin{array}{c} \\ \text{uncertainty} \\ \text{uncertain} \\ \text{deficit} \\ \text{congress} \\ \text{legislation} \\ \text{regulation} \\ \text{federal\_reserve} \\ \text{white\_house} \\ \text{economic} \\ \text{economy} \\ \vdots \end{array} \begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ \vdots & \vdots & \end{pmatrix} \end{array}$$

We note that $\hat{\boldsymbol{\Omega}}_{K=3}$ is exactly the EPU selection matrix, thus confirming that our algorithm can converge to an exact solution even with such a large search space.

In Table 1, we report the number of active tokens and the training window, validation window, and test window root mean-squared-errors (RMSE) for each epoch, along with the optimal results after the pruning step. We notice that the reduction in all error measures is much less pronounced when the wrong number of dimensions of the selection matrix is used. We also see that the pruning step was necessary for retrieving the exact EPU matrix when optimizing the three-dimensional selection matrix. Indeed, on the 13 epochs, the three-dimensional selection matrix had 11 active tokens, of which one was removed after pruning. Finally, we note that no active token is added for some epochs, but for subsequent epochs, more than one active token is added. This might be because a token is only more informative in the presence of another token and not by itself.

Table 1: Optimization process for the EPU index target

		Panel A: $K = 1$		
Epoch	#Active	Training	Validation	Test
1	1	49.94	58.07	51.73
2	1	49.94	58.07	51.73
3	3	47.78	56.97	53.61
4	3	47.78	56.97	53.61
5	5	43.73	55.64	60.66
6	5	43.73	55.64	60.66
7	5	43.73	55.64	60.66
8	5	43.73	55.64	60.66
9	5	43.73	55.64	60.66
10	5	43.73	55.64	60.66
11	5	43.73	55.64	60.66
12	5	43.73	55.64	60.66
13	5	43.73	55.64	60.66
14	5	43.73	55.64	60.66
15	5	43.73	55.64	60.66
Pruning	4	46.67	55.77	57.61

Epoch	#Active	Training	Validation	Test
		Panel B: $K = 2$		
1	2	50.92	56.55	43.74
2	2	50.92	56.55	43.74
3	4	44.88	56.20	45.60
4	5	44.47	55.40	46.98
5	6	43.49	55.21	47.64
6	7	41.83	53.41	47.16
7	8	41.35	53.23	47.28
8	9	39.51	48.06	54.53
9	10	38.67	47.66	54.44
10	10	38.67	47.66	54.44
11	12	33.36	41.73	53.69
12	12	33.36	41.73	53.69
13	12	33.36	41.73	53.69
14	12	33.36	41.73	53.69
Pruning	12	33.36	41.73	53.69
		Panel C: $K = 3$		
Epoch	#Active	Training	Validation	Test
1	3	56.57	62.70	47.57
2	4	49.35	57.40	48.58
3	5	47.65	53.26	47.71
4	6	47.13	52.45	47.02
5	7	42.78	46.36	45.10
6	8	41.11	43.90	44.90
7	9	33.72	37.51	40.81
8	10	11.37	12.26	17.34
9	11	0.00	0.00	0.00
10	11	0.00	0.00	0.00
11	11	0.00	0.00	0.00
12	11	0.00	0.00	0.00
13	11	0.00	0.00	0.00
Pruning	10	0.00	0.00	0.00

This table reports the number of active tokens, the training, validation, and test window root mean square error (RMSE, $\times 100$) for each optimization epoch. We report this for $K = 1, 2, 3$-dimensional selection matrix optimization.

The computational time on a standalone computer (single Intel core i9 3.7GHz with 48 GB RAM) ranges from five hours and 25 minutes for $K = 1$ to nine hours and 45 minutes for $K = 3$. Overall, this validation exercise demonstrates that our optimization can recover pre-defined selection matrices within a reasonable amount of computational time.

4.3 VIX and inflation expectations

Next, we test our methodology for (i) tracking the VIX index and (ii) nowcasting inflation expectations. We use those two usecases because news articles were previously used to build text-indices to forecast these variables. Thus, we have natural benchmarks to compare the performance of our approach.

For the VIX, Baker et al. (2019) construct the Equity Market Volatility (EMV) index, which, in our framework, can be defined with the following selection matrix:

$$\boldsymbol{\Omega}_{\text{EMV}} \equiv \begin{array}{c} \\ \text{economic} \\ \text{economy} \\ \text{financial} \\ \text{stock\_market} \\ \text{equity} \\ \text{equities} \\ \text{standard\_and\_poors} \\ \text{s\&p} \\ \text{volatility} \\ \text{volatile} \\ \text{uncertain} \\ \text{uncertainty} \\ \text{risk} \\ \text{risky} \end{array} \begin{array}{c} \begin{array}{ccc} \text{Equity} & \text{Market} & \text{Volatility} \end{array} \\ \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix} \end{array}$$

This news selection is then further analyzed and divided to quantify journalist perceptions about the news items, developments, concerns, and anticipations that drive volatility in equity returns. By optimizing the selection matrix, we aim to derive an attention index that more closely matches the VIX than the EMV does.[4] This is particularly useful as a better overarching index incorporates a more precise selection for selecting news having a positive relationship with market volatility, and thus better explains, in a quantitative manner (due to the availability of news data), the level of volatility. This could then be decomposed into a more narrow indicator as in Baker et al. (2019).

Regarding inflation expectations, Pfajfar and Santoro (2013) and Marcellino and Stevanovic (2022) analyze the link between news about inflation and how households form expectations about inflation. In particular, Marcellino and Stevanovic (2022) conclude that media communication and agents' attention play an important role in aggregate inflation expectations. In our framework, their attention index can be defined as:[5]

$$\boldsymbol{\Omega}_{\text{INFL}} \equiv \begin{array}{c} \\ \text{inflation} \end{array} \begin{array}{c} \begin{array}{c} \text{inflation} \end{array} \\ \begin{pmatrix} 1 \end{pmatrix} \end{array}$$

For a measure of inflation expectations, we use data from the Michigan University Survey of Consumers; we consider the median expected price change for the next 12 months (MICH).[6] Similar to the VIX, we want to find a selection matrix that generates a news attention index that more closely matches inflation expectations. This, in turn, can be used to determine better what news are related to inflation expectations by analyzing the tokens in the selection matrix or by further analyzing the news selected by the selection matrix (for instance, via a topic model).

The time-series indices constructed with $\boldsymbol{\Omega}_{\text{EMV}}$ and $\boldsymbol{\Omega}_{\text{INFL}}$ will serve as benchmarks when evaluating the fit on the test window. For the optimization, we use a population of size $Q = 200$ and $H = 500$ iterations per epoch with a maximum of 15 tokens (and, as such 13-15 epochs depending on the starting number of filtering dimensions). We set $\lambda_1 = 0.25$ and optimize for tokens positively correlated with the target index ($\lambda_2 = 0$ and $\lambda_3 = 1$). We keep the solution among $K = 1, 2, 3$ with the lowest objective value from the combined training and validation windows. Finally, we benchmark the results against $f_{\text{att}}(\boldsymbol{\Omega}_{\text{EMV}}, \boldsymbol{C}_t)$ and $f_{\text{att}}(\boldsymbol{\Omega}_{\text{INFL}}, \boldsymbol{C}_t)$ for the VIX and inflation expectations, respectivelly.

[4] The VIX is retrieved from https://fred.stlouisfed.org/series/VIXCLS.

[5] In particular Marcellino and Stevanovic (2022), search for instance of words containing "inflat" while Pfajfar and Santoro (2013) search for words whose root is "inflation." Since our words are lemmatized, all those instances are included in the word "inflation."

[6] MICH is retrieved from https://fred.stlouisfed.org/series/MICH.

The best selection matrices obtained on the combined training and validation windows after pruning are shown below:

$$\hat{\boldsymbol{\Omega}}_{\text{EMV}} = \begin{matrix} \text{economy} \\ \text{stock\_market} \\ \text{crisis} \\ \text{volatility} \\ \text{despite} \\ \text{club} \\ \text{fear} \\ \text{plunge} \\ \text{industrial\_average} \\ \text{package} \\ \text{summit} \\ \text{slash} \\ \text{antitrust} \\ \vdots \end{matrix} \begin{pmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \\ \vdots & \end{pmatrix} \qquad \hat{\boldsymbol{\Omega}}_{\text{INFL}} = \begin{matrix} \text{food} \\ \text{customer} \\ \text{frustration} \\ \text{hardware} \\ \text{vote} \\ \text{involved} \\ \text{disruption} \\ \text{clothes} \\ \text{suspend} \\ \text{shortage} \\ \text{gasoline} \\ \text{inflation} \\ \vdots \end{matrix} \begin{pmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \\ \vdots & \end{pmatrix}$$

A two-dimensional selection matrix generates the best performance in both cases. We see that most selected tokens by our optimization strategy have an intuitive relation with the target variable. This data-driven selection could be improved either (i) by augmenting the population size, the number of iterations, or the number of active tokens in the algorithm or (ii) by proceeding with a final human-supervised pruning step.

Table 2 reports the root mean squared error (RMSE) of linear regression models applied to tracking the VIX and inflation expectations using benchmarks and optimal indices across different time windows. As expected, the optimization process leads to an optimal index that exhibits lower RMSE for both the VIX and inflation expectations during the training and validation periods. The reduction in RMSE is substantial for the VIX (49% in the training window and 39% in the validation window) and inflation expectations (23% in the training window and 27% in the validation window). More importantly, the outperformance persists on the test window (data not observed during the selection matrix optimization process), reducing RMSE by 37% for the VIX and 25% for inflation expectations relative to the benchmarks.

Table 2: Performance results of selection matrix process for the VIX and inflation targets

	Training	Validation	Test
	Panel A: VIX		
Optimal	31.53	43.93	46.14
EMV	61.10	71.73	72.78
	Panel B: MICH		
	Training	Validation	Test
Optimal	52.64	64.28	66.16
Inflation	68.09	87.89	87.73

This table reports the root mean squared error (RMSE, ×100) measure (under various time windows) for the optimal selection matrix as well as for its benchmark for the VIX and MICH (twelve-month inflation expectations).

Further examination of the differences in performance is provided in Figures 1 and 2, which display the cumulative squared error difference between the benchmarks and the optimal indices. An upward-sloping curve indicates that the optimal selection matrix outperforms the benchmark. For the VIX, the outperformance of the optimal index remains relatively consistent during the test window, except for a notable flattening towards the end of the sample period, corresponding to a lower volatility period after

the initial spike due to the COVID-19 pandemic. For inflation expectations, the outperformance of the optimal index is evident from 2016 to mid-2018, followed by a slight underperformance until 2020. Interestingly, the optimal index exhibits superior performance as inflation expectations rise towards the end of the sample period.

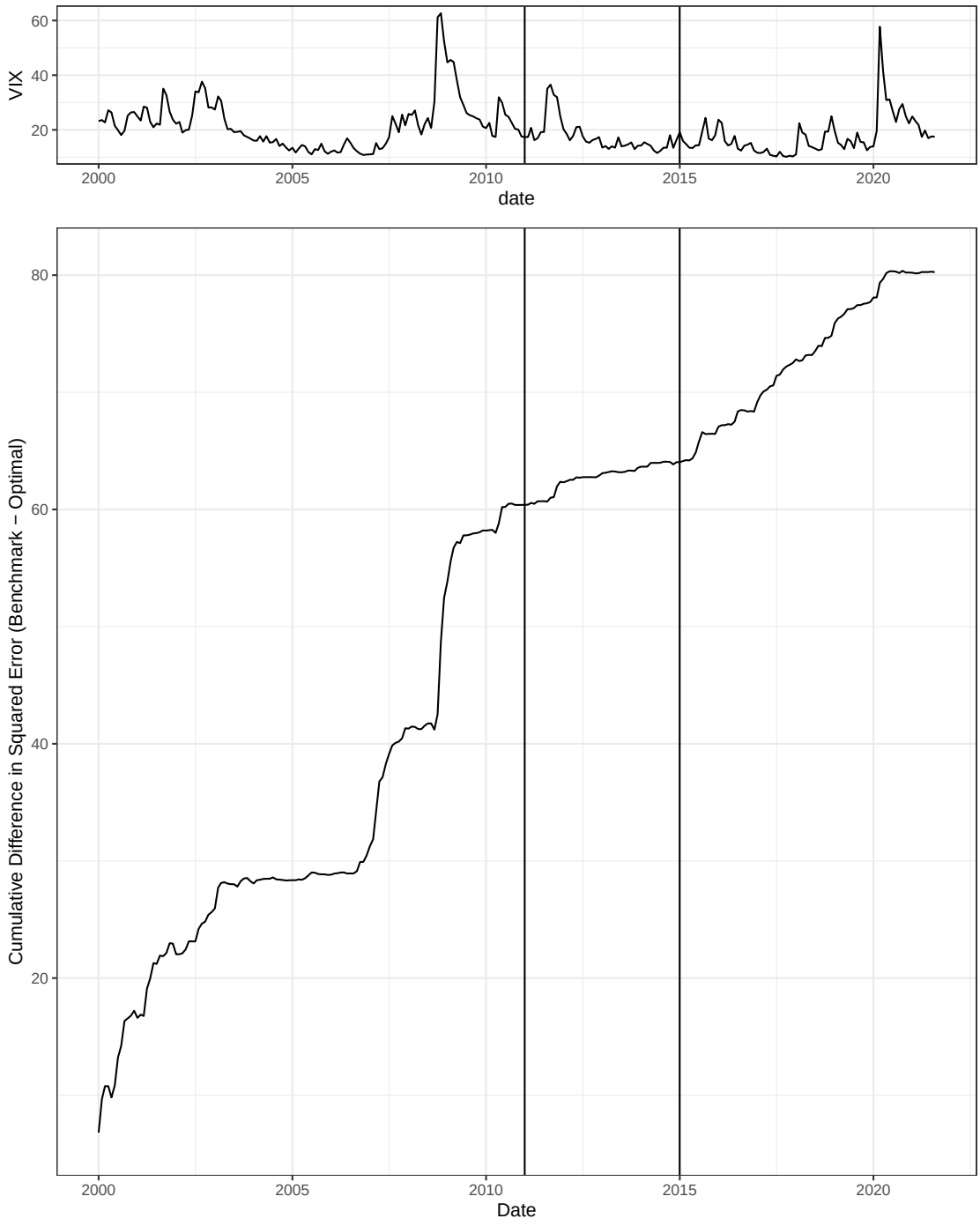

Figure 1: Cumulative squared error difference – VIX This figure shows the cumulative squared error difference for the results generated by the benchmark selection matrix minus the one generated by the optimal selection matrix for the VIX usecase. Vertical lines indicate the end of the training window and the validation window.

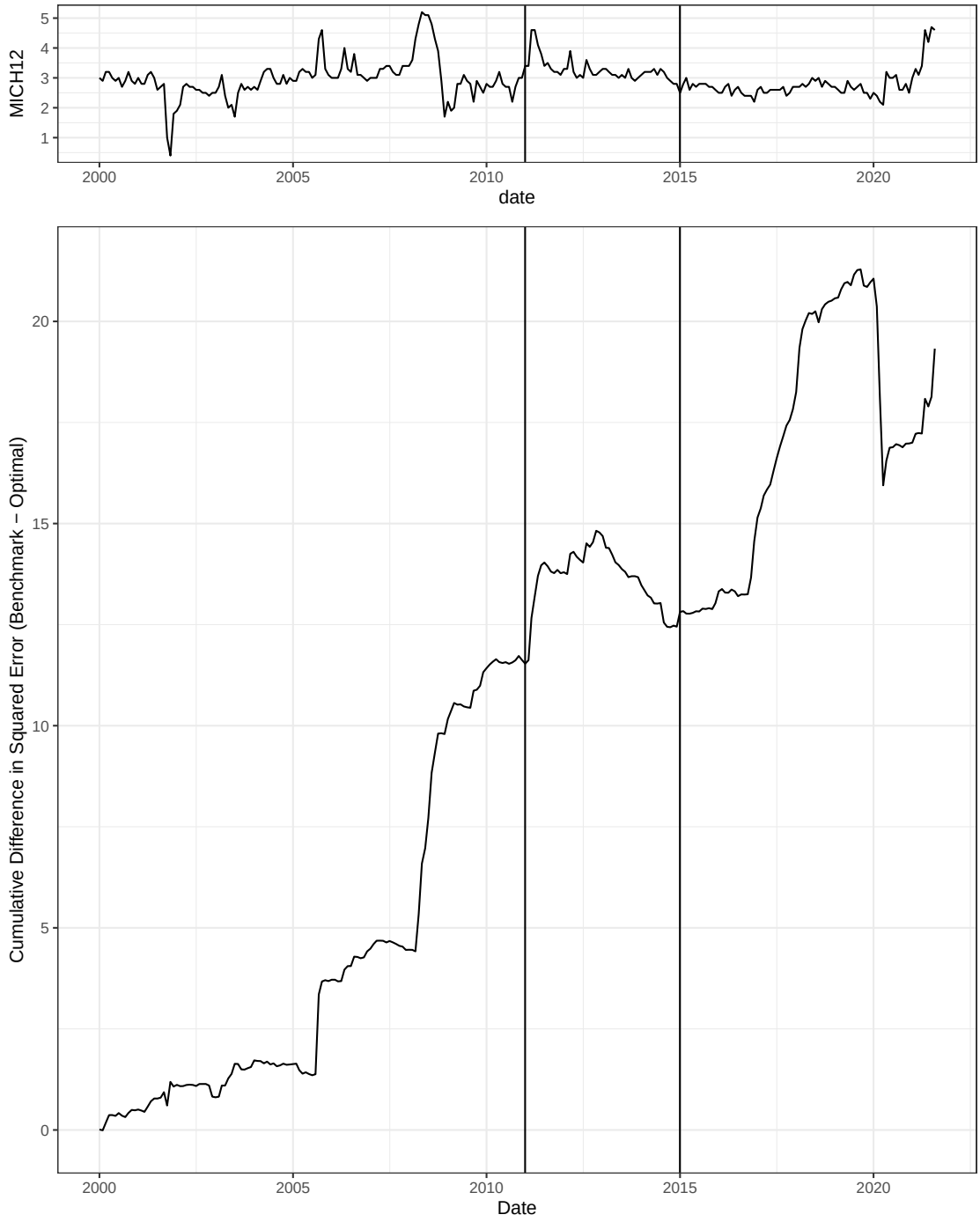

Figure 2: Cumulative squared error difference – MICH This figure shows the cumulative squared error difference for the results generated by the benchmark selection matrix minus the one generated by the optimal selection matrix for the MICH (twelve-month inflation expectation) usecase. Vertical lines indicate the end of the training window and the validation window.

5 Conclusion

In this research, we address a critical aspect often overlooked in economic and financial analysis using textual data—the text selection process. We introduce an algorithm that determine which set of texts, among a large corpus, leads to a text-based index that is optimal for a specific objective—typically, an index that maximizes the contemporaneous relation or the predictive performance with respect to

a target variable, such as inflation. Our approach, based on a genetic algorithm and tailored crossover and mutation operators, offers a data-driven and systematic text selection procedure.

We illustrate the relevance of our approach using a large collection of news articles from the Wall Street Journal. In particular, we show how to improve the existing news-based VIX index by Baker et al. (2019) or the news-based inflation expectations index by Pfajfar and Santoro (2013) and Marcellino and Stevanovic (2022).

References

Algaba, A., D. Ardia, K. Bluteau, S. Borms, and K. Boudt. 2020. Econometrics meets sentiment: An overview of methodology and applications. Journal of Economic Surveys 34:512–47.

Ardia, D., K. Bluteau, and K. Boudt. 2019. Questioning the news about economic growth: Sparse forecasting using thousands of news-based sentiment values. International Journal of Forecasting 35:1370–86. doi:10.1016/j.ijforecast.2018.10.010.

———. 2022. Media abnormal tone, earnings announcements, and the stock market. Journal of Financial Markets 61:100683–.

Ardia, D., K. Bluteau, K. Boudt, and K. Inghelbrecht. 2023. Climate change concerns and the performance of green vs. brown stocks. Management Science 69:7607–32. doi:10.1287/mnsc.2022.4636.

Ardia, D., K. Bluteau, and A. Kassem. 2021. A century of economic policy uncertainty through the French–Canadian lens. Economics Letters 205:109938–.

Ardia, D., K. Bluteau, and M. Meghani. 202x. Thirty years of academic finance. doi:10.1111/joes.12571. Journal of Economic Surveys, Forthcoming.

Baker, S. R., N. Bloom, and S. J. Davis. 2016. Measuring economic policy uncertainty. Quarterly Journal of Economics 131:1593–636. doi:10.1093/qje/qjw024.

Baker, S. R., N. Bloom, S. J. Davis, and K. J. Kost. 2019. Policy news and stock market volatility. Working Paper, National Bureau of Economic Research.

Caldara, D., and M. Iacoviello. 2022. Measuring geopolitical risk. American Economic Review 112:1194–225.

Gavriilidis, K. 2021. Measuring climate policy uncertainty. Working paper.

Gentzkow, M., B. Kelly, and M. Taddy. 2019. Text as data. Journal of Economic Literature 57:535–74.

Handley, K., and N. Limão. 2022. Trade policy uncertainty. Annual Review of Economics 14:363–95.

Hansen, S., M. McMahon, and A. Prat. 2018. Transparency and deliberation within the FOMC: A computational linguistics approach. Quarterly Journal of Economics 133:801–70. doi:10.1093/qje/qjx045.

Husted, L., J. Rogers, and B. Sun. 2020. Monetary policy uncertainty. Journal of Monetary Economics 115:20–36.

Jegadeesh, N., and D. Wu. 2013. Word power: A new approach for content analysis. Journal of Financial Economics 110:712–29.

Justeson, J. S., and S. M. Katz. 1995. Technical terminology: Some linguistic properties and an algorithm for identification in text. Natural Language Engineering 1:9–27. doi:10.1017/s1351324900000048.

Kelly, B., A. Manela, and A. Moreira. 2021. Text selection. Journal of Business & Economic Statistics 39:859–79.

Loughran, T., and B. McDonald. 2014. Measuring readability in financial disclosures. Journal of Finance 69:1643–71.

Manela, A., and A. Moreira. 2017. News implied volatility and disaster concerns. Journal of Financial Economics 123:137–62.

Marcellino, M. G., and D. Stevanovic. 2022. The demand and supply of information about inflation. Working paper.

Pfajfar, D., and E. Santoro. 2013. News on inflation and the epidemiology of inflation expectations. Journal of Money, Credit and Banking 45:1045–67.

Rahimikia, E., S. Zohren, and S.-H. Poon. 2021. Realised volatility forecasting: Machine learning via financial word embedding. Working paper.

Rose, S., D. Engel, N. Cramer, and W. Cowley. 2010. Automatic keyword extraction from individual documents. In Text mining: Applications and Theory, 1–20. John Wiley & Sons, Ltd. doi:10.1002/9780470689646.ch1.

Scrucca, L. 2013. GA: A package for genetic algorithms in R. Journal of Statistical Software 53. ISSN 1548-7660. doi:10.18637/jss.v053.i04.

———. 2017. On some extensions to GA package: Hybrid optimisation, parallelisation and islands evolution. R Journal 9:187–. doi:10.32614/rj-2017-008.

Wijffels, J. 2021. udpipe: Tokenization, parts of speech tagging, lemmatization and dependency parsing with the 'udpipe' 'nlp' toolkit.