Journal of Data Science 22 (3), 409–422
July 2024

DOI: 10.6339/24-JDS1147
Data Science in Action

Bringing Search to the Economic Census – The NAPCS Classification Tool☆

Clayton Knappenberger[1]
[1]*U.S. Census Bureau, 4600 Silver Hill Road, Washington, DC 20233, USA*

Abstract

The North American Product Classification System (NAPCS) was first introduced in the 2017 Economic Census and provides greater detail on the range of products and services offered by businesses than what was previously available with just an industry code. In the 2022 Economic Census, NAPCS consisted of 7,234 codes and respondents often found that they were unable to identify correct NAPCS codes for their business, leaving instead written descriptions of their products and services. Over one million of these needed to be reviewed by Census analysts in the 2017 Economic Census. The Smart Instrument NAPCS Classification Tool (SINCT) offers respondents a low latency search engine to find appropriate NAPCS codes based on a written description of their products and services. SINCT uses a neural network document embedding model (doc2vec) to embed respondent searches in a numerical space and then identifies NAPCS codes that are close to the search text. This paper shows one way in which machine learning can improve the survey respondent experience and reduce the amount of expensive manual processing that is necessary after data collection. We also show how relatively simple tools can achieve an estimated 72% top-ten accuracy with thousands of possible classes, limited training data, and strict latency requirements.

Keywords *natural language processing; neural networks; search; survey collection*

1 Introduction

Statistical agencies have long understood the need to automatically classify open-ended responses into predefined categories. Allowing respondents to provide open-ended text when answering questions reduces the amount of burden that respondents face trying to interact with complex classification systems. It also allows statistical agencies to collect data even in hard-to-classify cases. However, this approach comes at the cost of additional work for survey analysts who must manually code these responses.

One of the earliest efforts to automatically classify open-ended responses comes from O'Reagan (1972) who used keywords in the respondent-provided text to automatically assign Standard Industrial Classification (SIC) codes in the 1967 Economic Census. Other examples include Chen et al. (1993)'s work for the 1990 Decennial Census, and more recent approaches such as the Bureau of Labor Statistics autocoder for the Survey of Occupational Injuries and Illnesses (United

☆Any opinions and conclusions expressed herein are those of the author(s) and do not reflect the views of the U.S. Census Bureau. The Census Bureau has reviewed this data product to ensure appropriate access, use, and disclosure avoidance protection of the confidential source data (Project No. P-7504847), Disclosure Review Board (DRB) approval number: CBDRB-FY23-EWD001-002.

Received November 30, 2023; Accepted July 5, 2024

States Bureau of Labor Statistics, 2023) and the United Kingdom's Office of National Statistics' automated text coding for the 2021 Census (Office of National Statistics, 2023). The Economic Census is conducted every 5 years and collects data from over 8 million U.S. businesses. The Census Bureau can collect data for many small businesses directly from administrative records, but in 2022, approximately 4.2 million business establishments were asked to complete the Economic Census and report information using both the North American Industry Classification System (NAICS) and the North American Product Classification System (NAPCS) (United States Census Bureau, 2022).

NAPCS is a market or demand-based classification system for goods and services. It is meant to cross industries, meaning that a company operating in a single NAICS code can report multiple NAPCS codes and a single NAPCS code can be reported by companies across multiple industries. Questions relating to NAPCS were introduced in the 2017 Economic Census. Businesses were asked to report their total sales, shipments, receipts, or revenue broken down by NAPCS code (Wiley and Whitehead, 2022). This replaced the previously collected Product Lines data with a classification system that is standardized between Canada, Mexico, and the United States. The 7,234 NAPCS codes complement the existing roughly 1,000 NAICS codes because businesses can report revenue in multiple NAPCS codes instead of just their Primary Business or Activity (PBA) such as when reporting with NAICS. However, the large number of NAPCS codes and the ability to enter more than one increases the burden on respondents. Approximately one million NAPCS write-ins were submitted in the 2017 Economic Census (Wiley and Whitehead, 2022) because of the difficulty of identifying the correct NAPCS codes. With limited time and resources, analysts were only able to recode about 17% of these write-ins (Wiley and Whitehead, 2022). The Smart Instrument NAPCS Classification Tool (SINCT) was developed to reduce the burden that respondents experience reporting NAPCS data while also reducing the number of open-ended responses that analysts would need to recode.

Respondents directly interacted with SINCT online from the Economic Census questionnaire instrument to retrieve NAPCS codes based on their previously reported NAICS and any search terms they provide. From the respondent's perspective, SINCT needs to be a seamless part of the experience, providing a low-latency search engine for picking appropriate NAPCS codes. This is in sharp contrast to many prior research efforts, which devised systems for automatically coding text data after it had already been collected. In these prior systems, the respondent did not interact the automatic coding system and likely had no idea their text response would later be converted into a classification code. Consequently, there were strict latency requirements that SINCT return search results in less than 4 seconds. If a search exceeded the 4 second threshold, the questionnaire web page would display a screen with no results.

Successfully meeting these requirements, SINCT is an improvement over previous automatic coding tools in several ways. First, by allowing respondents to directly interact with the machine learning predictions and choose the classification codes that best match their written descriptions in real time, SINCT keeps the respondent in-the-loop and reduces the need for analysts to validate machine learning predictions. Other methods that incorporate user feedback do so only when the model is unsure (Moscardi and Schultz, 2023), making it more difficult to determine when the model's performance is beginning to degrade. Second, by allowing respondents to select all that apply, SINCT encourages reporting at a finer level of detail, reducing the need for analysts to manually split combined or vague searches. Finally, SINCT is lightweight and fast. It shares server resources with other machine learning tools and still returns search results in 0.1 seconds on average.

2 Related Work

Automatic coding of open-ended survey responses has a long history at the U.S. Census Bureau. Early work drew inspiration from information retrieval (IR) methods and focused on building reference files of unigram and bigram co-occurrences with the desired classification codes (O'Reagan, 1972). IR methods continue to be prevalent in automatic coding, but modern applications diverge in how they use these ideas. An earlier version of SINCT utilized term frequency-inverse document frequencies (TF-IDFs) to retrieve relevant NAPCS codes to a user's query (Wiley and Whitehead, 2022). Dumbacher and Whitehead (2024) took a more sophisticated approach of developing an ensemble of IR methods for recommending NAICS codes to respondents in the 2022 Economic Census.

Several researchers trained machine learning models in their applications. Roberson and Nguyen (2018) found that training a regularized logistic regression on term frequency features yielded the best performance across a range of metrics for automatically classifying written descriptions from the Annual Capital Expenditures Survey. Moscardi and Schultz (2023) extended this idea for the Commodity Flow Survey by training their regularized logistic regression on TF-IDF features. Roberson (2021) used linear support vector machine (SVM) combined with latent semantic analysis (LSA) on the term frequency matrix of unigrams to correctly predict the NAPCS code of a product description 96% of the time for a subset of 44 NAPCS codes.

Many of these methods are currently being used in production settings and respondents can directly interact with them while filling out their surveys. Tools existing within the survey introduce new possibilities as well as new challenges. On the one hand, if a survey respondent can directly search for the NAPCS codes corresponding to the products and services they offer, this means that respondent chooses which NAPCS codes get assigned and not an automatic coding system. Ideally, this will improve data quality and reduce respondent burden. On the other hand, this change requires researchers to rethink how we evaluate these models and the trade-offs involved. For instance, few things are more likely to frustrate a survey respondent than having to wait for a web page to load, so models need to meet strict latency requirements to avoid degrading the respondent's experience.

Perhaps because of this stricter latency requirement, developments in automatic coding have not advanced with developments in natural language processing (NLP). Since the introduction of word2vec (Mikolov et al., 2013a,b), neural network-based embedding models have dominated NLP. The skip-gram architecture of Mikolov et al. (2013a) is quite simple – feed a one-hot encoding of a single word through a fully connected neural network with a single hidden layer to predict what words surround this work in the training data. After training, the hidden layer's weights work as a numerical representation of individual words (i.e., word embeddings). Le and Mikolov (2014) extended this idea to training document embeddings either individually or jointly with word embeddings. The Python library Gensim (Řehůřek and Sojka, 2010) popularized this algorithm as *doc2vec*. Among the improvements offered by doc2vec is that it can encode variable length sequences into a fixed dimension vector – enabling easy comparison of documents simply by calculating the cosine similarity of their embeddings.

The emergence of neural network-based embeddings meant that learned weights began to replace traditional IR methods. These weights could be fine-tuned to specific tasks like downstream classification. Measure (2017) takes advantage of this in their Survey of Occupational Injuries and Illnesses (SOII) automatic coder. The SOII automatic coder uses the NAICS code for the business along with several text fields detailing different aspects of the workplace injury or illness incident to classify an incident into the Standard Occupational Classification (SOC) system and the Occupational Injuries and Illnesses Classification System (OIICS). Their neural network architecture uses a sequence of convolutional (LeCun et al., 1990), highway (Srivastava et al., 2015), and bidirectional long short-term memory layers (LSTM) (Graves and Schmidhuber, 2005) to generate word and document embeddings (which they call field encodings) for each of the text fields collected in their survey. The field encodings are then passed through an attention layer to yield a combined document encoding that is concatenated with separately learned NAICS embeddings. These pass through two additional highway layers and, finally, a fully connected softmax classification layer to predict the probability of each classification code.

Vaswani et al. (2017) introduced the Transformer architecture which was subsequently improved by Devlin et al. (2019)'s Bidirectional Encoder Representations from Transformers (BERT). These researchers introduced a much simpler architecture for fine-tuning word embeddings. They suggest that a practical use-case is to pretrain BERT in an unsupervised fashion on a large collection of text, and then afterwards, to add a softmax classification layer at the end of a transformer block and fine-tune the model weights on labeled data. The U.S. Bureau of Labor Statistics currently uses Transformers for automatic coding in production (United States Bureau of Labor Statistics, 2023), running the model offline to automatically code text descriptions after the respondent has submitted their survey. Roberson (2021) rightly points out the difficulties in using these models for training or inference without graphical processing units (GPUs). Agencies that do not have ready access to GPUs will continue to struggle in building applications that are both accurate and have sufficiently low latency to reliably work as a search engine within a survey.

Researchers have suggested multiple ways to evaluate automatic coding systems. Chen et al. (1993) evaluated their automatic coding system using two metrics. The production rate is the number of cases that the automatic coding system is able to classify divided by the total number of cases, while the error rate is the number of cases where the automatically assigned code does not match the code manually assigned by coding experts. Büttcher et al. (2016) offer two different evaluation criteria for information retrieval systems: efficiency and efficacy. Efficiency is often measured by the latency of the system – how long a user must wait between querying the system and receiving their results. Efficacy is often measured with Precision at K which is defined as the number of results relevant to the user in the first K search results. Both the production and error rates (Chen et al., 1993) assume that for a given text description there is only one correct answer and that the automatic coding system is only able to return zero or one result for each text description. In contrast, Precision at K assumes that for a single text description there are multiple correct codes for a single text description and that a user wants to see as many of these in the first K results as possible.

Unfortunately, the commonly used evaluation methods suggested by Chen et al. (1993) and by Büttcher et al. (2016) are not easy to apply to SINCT. As will be more fully explained in Section 4, SINCT recommends ten NAPCS codes for every respondent search text, and the respondent can choose multiple NAPCS codes from that list of ten. SINCT assists the user rather than replaces them. This makes the production rate inapplicable as an evaluation method. We can estimate an error rate using a held-out test dataset, but only using the first result from SINCT makes this estimate too conservative. In production, the respondent was able to select any of the ten items from the list. Likewise, it is difficult to estimate SINCT's Precision at K since the test data available were manually coded by experts who only assigned a single NAPCS code to each write-in. In Section 5 we blend ideas from both Chen et al. (1993) and Büttcher et al. (2016) to define two efficacy metrics.

3 Data

The key to each of the successful attempts at building automatic coders referenced in Section 2 was the availability of labeled training data. In each case, researchers had access to a large corpus of labeled examples that had been manually coded by supporting staff. In this respect, SINCT is no different. NAPCS was first used by the U.S. Census Bureau in the 2017 Economic Census; prior to that the 2012 Economic Census used a different product classification system called Product Lines. In both Economic Censuses, respondents were asked to report their total revenue broken down into product codes (Product Lines in 2012 and NAPCS in 2017). Respondents also had the option to report revenue in an "Other" category and to provide a written description of the product or service (Wiley and Whitehead, 2022). Census staff manually reviewed these write-ins and reclassified them into a NAPCS code. In the 2017 Economic Census, respondents left about one million NAPCS write-ins, but Census staff were only able to manually reclassify about 17% of these due to limited time and resources (Wiley and Whitehead, 2022). Additionally, not every write-in could be reclassified, since the written descriptions did not always contain enough detail to apply to a single NAPCS code.

Table 1 updated from Wiley and Whitehead (2022) summarizes each of the datasets used to train SINCT. First, combining all the reclassified write-ins from the 2012 and 2017 Economic Censuses yielded a dataset of approximately 180,000 examples. Second, the Classification Analytical Processing System (CAPS) is a database of examples drawn from analyst review for other analysts to use as they review write-ins. Third, is a dataset of respondent-provided searches pulled from the 2021 Refile Field Test to the 2022 Economic Census that were manually coded to NAPCS codes by Census reviewers. Finally, the NAPCS title file was used because it provided at least one example for every valid NAPCS code. Economic Census write-ins, CAPS, and the 2021 Refile Field Test datasets all had to be updated to the 2022 NAPCS vintage using publicly available concordance files. Combined these datasets provided a training dataset of approximately 225,000 samples. About 4,500 samples were considered by Census Subject matter Experts (SMEs) as the most reliable, so we reserved these as test data.

NAPCS collection codes are a hierarchical system with two levels: broad line codes and detailed line codes. Broad line codes are ten-digit codes ending in 00. Detailed line codes are similarly ten-digit codes, but in contrast to broad line codes, detailed line codes end in something other than 00 and provide greater specificity within a broad line. For example, NAPCS code 7003825000 is a broad line code representing "Room or unit accommodation for travelers" while 7003825003 is a detailed line NAPCS code falling within the broad line code 7003825000 and representing "Room or unit accommodation for travelers, with maid service." Census SMEs reviewed the 7,234 NAPCS codes and determined that SINCT should only recommend NAPCS codes from a smaller list of 6,284 unique NAPCS codes to users. In some cases, SMEs asked SINCT to only recommend the broad line, but in other cases SMEs decided that SINCT should recommend the detailed lines and not the associated broad line. Reducing the number of valid NAPCS codes that SINCT could recommend eliminated approximately 75,500 training samples corresponding to these NAPCS codes. The vast majority of the dropped training samples came from reclassified write-ins from the 2012 and 2017 Economic Censuses. About 25% of the test cases had NAPCS codes that were dropped from our list of valid codes. Given our already smaller test sample, we did not wish to drop these cases from our test data and instead chose to relax our evaluation criteria to allow any NAPCS recommendations that agreed with the test case at the broad line level to be counted as a success. Even after doing this, approximately 4% of the test cases had a NAPCS code whose broad line was not on the list of NAPCS codes SINCT

Table 1: Summary of SINCT data source advantages and disadvantages (Wiley and Whitehead, 2022).

Data Source	N.	Advantages	Disadvantages
Economic Census	180,000	Represents target population Reflects natural language	Descriptions not classified perfectly Descriptions contain misspellings
Classification Analytical Processing System (CAPS)	24,000	Provides a rich vocabulary Descriptions are classified correctly	Text does not always reflect natural language
Subject matter expert (SME) examples	11,500	Incorporates institutional knowledge Can be continuously updated with SME feedback	Small data source
NAPCS title file	9,100	Definitive source of NAPCS descriptions	Small data source

Source: 2012 & 2017 Economic Census; Classification and Analytical Processing System; Census Bureau Analysts; and https://www.census.gov/naics/napcs; all counts are rounded.

was allowed to recommend. This implies that SINCT's maximum accuracy was 96% on this test sample and reduced our training sample down to approximately 145,000 training samples and 4,500 test samples.

The NAPCS system is both more detailed and has fewer labeled training samples than similar automatic coding systems developed by other researchers. Instinctively it makes sense that the success of a learning algorithm depends not just on the number of samples, but also on the number of samples per class. Table 2 summarizes the data available for six recent efforts at performing automatic coding described in Section 2, ordered by the number of samples divided by the number of target classes (codes in their respective classification systems). SINCT ranks at the bottom of this list, having only about 23 samples per target class, compared to 318 samples per target class in Roberson (2021)'s NAPCS research. Real data are not usually evenly divided in this way, but this simple calculation helps to illustrate the problem. Almost 20% of the valid NAPCS codes that SINCT needs to be able to recommend have no respondent-provided training examples at all.

4 Model

One of the oldest ideas in information retrieval is the Vector Space Model, where queries and documents are represented in a high dimensional vector space where each vector corresponds to a specific term in the corpus vocabulary (Büttcher et al., 2016). Retrieval and ranking of documents that are relevant to a user query is as simple as calculating the similarity (usually cosine) between the user's query and the documents in the vector space. If we retrieve only the

Table 2: Summary of data and methods for prior automatic coding research.

Researcher	Classification System	Method	N. Codes	N. Samples
Roberson and Nguyen (2018)	ACES (Equipment, Structures, and Not Applicable)	Logistic Regression on term frequencies	3	14,000
Dumbacher and Whitehead (2024)	NAICS	Hierarchical ensemble of IR methods with learned weights	1,012	3.7 million
Measure (2017)	SOC/OIICS	Convolutional embeddings and LSTM recurrent deep neural network on text descriptions and NAICS	1,400/800	1.3 million
Moscardi and Schultz (2023)	SCTG	Logistic Regression on unigram, bigram, 3-5 character n-grams TF-IDF weighted term frequencies and one-hot-encoded NAIC codes	514	400,000
Roberson (2021)	NAPCS	Support Vector Machine on term frequencies	44	14,000
SINCT	NAPCS	Nearest Neighbor retrieval on Doc2Vec trained embeddings	6,284	145,500

Source: see cited sources for information about their data sources and Table 1 above for information about SINCT's data sources.

most similar document, this is equivalent to a k-Nearest Neighbors classifier with one nearest neighbor and cosine similarity being used as the similarity measure. Hastie et al. (2011) suggest that k-Nearest Neighbors classifiers tend to be successful when the decision boundary separating classes is very irregular. Mitchell (1997) agrees that k-Nearest Neighbors classifiers are effective in many problems but points out that k-Nearest Neighbors classifiers suffer from the curse of dimensionality because irrelevant features in the vector space can make similar documents appear farther apart. Mitchell (1997) additionally points out that because k-Nearest Neighbors classifiers postpone all processing until prediction, they can take longer to make predictions than other classifiers as they must search through all n training documents.

Nearest neighbor search lies at the heart of SINCT, and we address the issue of longer prediction times by using dense learned document embeddings. Word2Vec (Mikolov et al., 2013a,b) was developed to learn a vector space representation of words and differs from the Vector Space Model in two respects. First, a point in the Vector Space Model provides a numerical representation for a document comprised of a set of words whereas a point in a word2vec emedding represents a single word.

Second, word2vec embeddings are learned by a neural network from the Vector Space Model. The skip-gram architecture introduced in Mikolov et al. (2013a) is rather simple. It accepts a single word initially encoded using the Vector Space Model, and this encoding is then fully connected to an output layer through a single hidden layer. The output layer uses either hierarchical softmax (Mikolov et al., 2013a) or negative sampling (Mikolov et al., 2013b) to predict what words typically surround the input word. After training this neural network, the weights of the hidden layer serve as word embeddings. Because the size of the hidden layer H is significantly smaller than the size of the corpus' vocabulary V, these embeddings are a denser representation of the word than those offered by the Vector Space Model. These word embeddings retain much of the relationship between words, and it is easy to identify words with similar meanings in the corpus by performing a nearest neighbor search on the word embeddings.

An obvious downside to Word2Vec is that SINCT's search task is to retrieve documents rather than words. This challenge is easily overcome by replacing the word encoding input from the skip-gram architecture with a vector of document IDs, and then using this document encoding to predict a sequence of randomly drawn words from the document. This approach is called the distributed bag of words (DBOW) architecture in Le and Mikolov (2014). Analogously with Word2Vec, the trained hidden layer weights are used to represent documents and similarity between documents can be computed via their cosine similarity in the embedded document space. Documents which are close to each other in this space have similar meanings to each other. One complication to using the DBOW architecture for document embeddings is that in production, new user queries will not have been seen by the model during training, and hence a new document ID must be added to the input layer of document encodings and a new document embedding must be learned via gradient descent. During this inference step, the previously learned hidden layer and softmax layer weights are fixed (Le and Mikolov, 2014). This has the consequence that SINCT must first embed queries in the document space before retrieving similar documents from the training corpus, adding to SINCT's computational complexity.

SINCT uses the Python library Gensim, which offers a fast and efficient implementation of the DBOW model which they call Doc2Vec in homage to the original Word2Vec model (Řehůřek and Sojka, 2010). Gensim implements several parameters for training a Doc2Vec model, consistent with Le and Mikolov (2014), but with two new features:

- A learning rate *alpha* which by default will linearly decrease over training to the value of *min_alpha* so that later training epochs will have a smaller impact on the learned weights and;
- The ability to learn word vectors in skip-gram fashion simultaneously with the DBOW architecture using the *dbow_words* parameter.

Table 3 shows some of the different parameters available for training a Doc2Vec model, the default Gensim setting, and what we chose for SINCT. We found that the most effective model parameters to be: a somewhat smaller than default vector size; a larger number of training epochs; and word vectors trained alongside the documents vectors. In general though, SINCT was robust to most of these parameters. Considering that the original vocabulary had tens of thousands of unique tokens, compressing the embedding space down to 60 dimensions mitigated the curse of dimensionality while also reducing the cost of performing nearest neighbor searches.

In short, SINCT has been a simple and lightweight model. Users interact with SINCT directly from the survey web page. Respondents are asked to select NAPCS codes from a provided list along with a search box where they can describe what products and services their business provides in natural language. After a user types something into this search box, the text and their NAICS code are sent to SINCT, which then loads a new screen on the respondent's web page showing a list of ten NAPCS codes related to their search. The respondent is free to choose as many NAPCS codes from this list as they want, with the option to perform additional searches based on different text inputs. If a respondent is unable to find a NAPCS code that matches their business, the respondent has the option to submit the text they searched for as a write-in. Census staff will need to analyze these write-ins later to determine if they can be coded into a NAPCS code. SINCT's speed is due to having only two steps to generate a set of recommendations for the respondent. First, the respondent's search is embedded into the same numerical space as the training data. Second, matrix multiplication is performed on the training data to calculate the cosine similarity of the search to each training example. After these two

Table 3: Doc2Vec training parameters.

Parameter	Meaning	Default	SINCT
vector_size	Dimensionality of the feature vectors	100	60
window	The maximum distance between the current and predicted word in a sequence	5	5
alpha	The initial learning rate	0.025	0.025
min_alpha	Learning rate will drop linearly to min_alpha during training	0.0001	0.0001
min_count	Ignores all words with total frequency lower than min_count	5	1
epochs	Number of training iterations over the corpus	10	50
negative	If > 0, negative sampling is used, the int for negative indicates how many noise words should be drawn	5	5
dbow_words	If set to 1 train word-vectors (in skip-gram fashion) simultaneously with DBOW doc-vector training; else only train doc-vectors	0	1

steps, ten NAPCS codes corresponding to training samples that are closest to the respondent's search are returned to the respondent.

5 Testing SINCT Variations

Prior to the mailout of the 2022 Economic Census, the U.S. Census Bureau tested three versions of the SINCT model based on SME feedback. Some SMEs were concerned that SINCT would rely too much on the NAICS code when recommending NAPCS codes. Establishments are supposed to be able to select NAPCS codes regardless of their industry code. These SMEs suggested that SINCT should support this by ignoring industry information when recommending NAPCS codes. Other SMEs felt that while in principle establishments could select NAPCS codes outside of their industry, in practice NAICS codes would be highly correlated with the relevant NAPCS codes for an establishment.

This second group of SMEs were also concerned that if SINCT did not use the industry information, then its recommendations would diverge widely from what the respondent intended. They felt that respondents were unlikely to provide sector specific keywords like "wholesale", "manufacturing", or "retail" in their SINCT searches and that lacking this information, SINCT might recommend manufacturing codes to wholesalers and vice versa.

Based on this feedback, the U.S. Census Bureau tested three different versions of the SINCT model.

1. Search only: SINCT used only the respondent-provided text to recommend NAPCS codes.
2. Search + NAICS: SINCT combined the respondent-provided text with the title of the NAICS code the respondent selected earlier in the questionnaire to recommend NAPCS codes.
3. Combined: SINCT ran the above two searches, generated two sets of recommendations, then combined and sorted these two lists by their cosine similarity.

Each version of the model was trained on the full training dataset of approximately 149,500 manually coded write-ins and then evaluated on the held-out test dataset of about 4,500 manually coded write-ins. We developed two efficacy metrics by drawing upon Chen et al. (1993) and Büttcher et al. (2016).

1. Search-level accuracy: the number of searches where the SME-chosen NAPCS code was in SINCT's top ten recommended NAPCS codes divided by the total number of searches. This answers the question: how likely is a search to return at least one correct NAPCS code to the user?
2. Respondent-level accuracy: the number of respondents with at least one search where SINCT recommended a correct NAPCS code divided by the total number of respondents. This answers the question: how likely is a user to have at least one successful search?

Both of these accuracy metrics relate to Chen et al. (1993)'s error rate by defining success or failure as binary (i.e. a search was or was not successful). They also relate to Büttcher et al. (2016)'s precision at K by counting a search as successful if the SME-chosen NAPCS code is anywhere in the list of the top-ten results. Additionally, both accuracy metrics are defined at the broad line level, meaning that if SINCT recommended a detailed line code and the respondent picked a different detailed line within the same broad line NAPCS code, we counted that as a successful recommendation. We measure efficiency using the inference latency which is defined as the number of seconds required for SINCT to return its top-ten recommendations after the respondent hits the search button.

6 Results

Table 4 summarizes the results of training these three separate SINCT models and then evaluating their performance on the held-out test data using our evaluation metrics. In general, the Search + NAICS model performed best, though at a slightly increased latency relative to the Search Only model.

Empirically, it seems that respondents chose NAPCS codes that at least somewhat aligned with their NAICS selections as the Search + NAICS model did better than both the Search Only and Combined models in the respondent-level and search-level accuracy metrics. The results in Table 4 show that approximately 86.1% of respondents were shown at least one correct NAPCS code across all of their searches using the Search + NAICS model. For individual searches, the

Table 4: SINCT accuracy and latency test results.

Metric	Search Only	Search + NAICS	Combined
Top-Ten Accuracy			
Respondent-level accuracy	74.7%	86.1%	76.2%
Search-level accuracy	61.7%	72.0%	63.3%
Inference Latency (s)			
Mean	0.013	0.014	0.025
Minimum	0.004	0.005	0.014
Median	0.011	0.012	0.025
Maximum	0.060	0.072	0.124

Source: see Table 1 for information about SINCT data; testing done September 12, 2022.

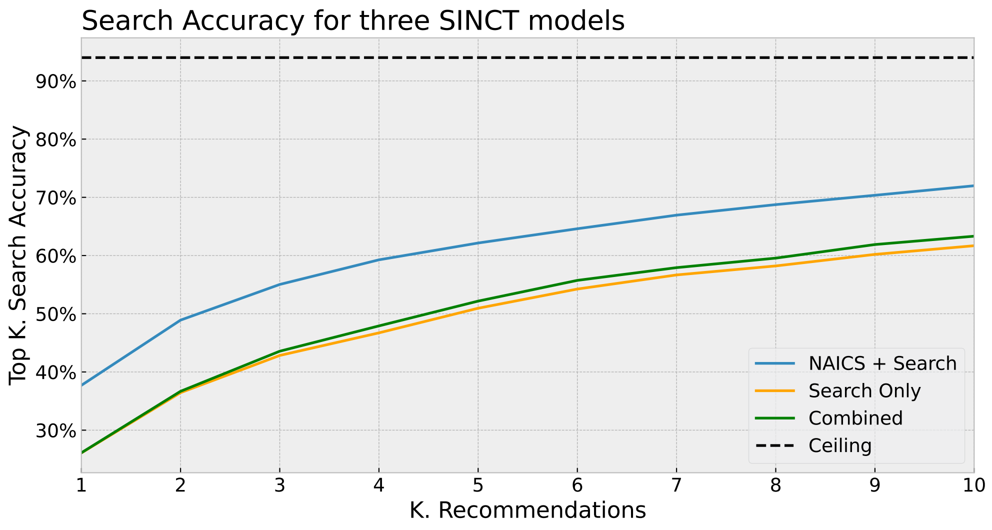

Figure 1: SINCT Accuracy Comparison for three models; See Table 1 for information about SINCT data; results from August 25, 2022.

Search + NAICS model returned at least one correct NAPCS code 72.0% of the time. This is about a ten percentage point improvement over an earlier version of SINCT reported by Wiley and Whitehead (2022). The next best performing model was the Combined model. However, as is clear from Figure 1, the combined model was only effective to the extent that recommendations from the Search + NAICS model were able to make it into the top ten recommended NAPCS codes.

In Figure 1 the top-one accuracy for all three models was under 40% but the top-one accuracy of the Search + NAICS model was approximately ten percentage points higher than both the Combined model and the Search Only model. Top-one accuracy corresponds closely to Chen et al. (1993)'s error rate – if the top-one accuracy of the NAICS + Search Model is about 39% then the corresponding error rate is 61%. The dashed line labeled "Ceiling %" represents the 96% of test cases that have valid NAPCS codes at the broad line level for SINCT to recommend from, the remaining 4% of the test cases remain as an irreducible error because they are not in the list of NAPCS codes SINCT is permitted to recommend (as described in Section 3).

The Search + NAICS version of the model performed best based on both accuracy metrics, but the model performed better for some sectors of the economy than others. For instance, if we only look at mining sector test cases, the Search + NAICS model achieves approximately 92.5% accuracy. However, if we examine the construction sector test cases, the same model only achieved a top-ten accuracy of 38.5%. For the other sectors present in the test data (i.e., services, manufacturing, retail, transportation, wholesale, and utilities) the model achieved a top-ten accuracy between 60 and 80%.

We believe that this variation in the model's performance is related to differences in the NAPCS codes for each sector. Construction, for instance, is particularly hard to predict because the NAPCS code a company chooses is driven by the kind of building the company is working on. A construction company working on nonresidential structures would have different NAPCS codes than if the same company worked on residential structures. Such detailed building information is not typically included in text descriptions and so the model is much less effective at recommending NAPCS codes to construction company respondents. Mining on the other hand is comparatively easier for SINCT given that there are only 88 mining NAPCS codes to predict from.

The results shown in Table 4 reflect a test environment with a single user and limited memory and CPU resources. These conditions differ from the field environment of the 2022 Economic Census where multiple respondents expect to use the tool at the same time and SINCT ran on a server with more memory and CPU resources. Consequently, Table 4 is useful for ranking the three models against each other, but do not conclusively show how well SINCT would perform in a field environment. The combined model was about twice as slow as the other two, because it had to call the other two models with each query. The Search + NAICS model and Search Only model had comparable latency, but this testing suggests the Search + NAICS model was slightly slower. A likely cause for this difference is simply that expanding the query to include the NAICS title also increased the number of words in the document to pre-process and embed.

Further performance testing was done on the Search + NAICS model on a test server with half of the resources available on the production server. This testing estimated that the production environment could sustainably handle 88 requests per second with SINCT returning results in about 1.2 seconds on average, well below our requirement that searches be completed in less than 4 seconds. Being able to sustainably handle 88 requests per second is also about 9 times the expected maximum throughput, which implies that in production SINCT returned results in a little over 0.1 seconds on average. For the entire period when the 2022 Economic Census was being collected, we did not receive a single complaint about SINCT timeouts. Additionally, compared to the 2017 Economic Census, the 2022 Economic Census received 78% fewer write-ins, further suggesting SINCT was effective compared to previous methods.

7 Conclusions

While there has been a great deal of research into automatically coding different classification systems based on respondent-provided written descriptions, most of this work has been done after collection and in bulk processing systems. It has only been recently that U.S. statistical agencies have been able to build tools within questionnaire web pages to assist respondents as they fill out their surveys. This advance in survey infrastructure reduces respondent burden and increases data quality but introduces new technical challenges. Namely, respondents expect any tool they interact with to be low latency. Additionally, prior work in automatic classification systems has been able to take advantage of rich datasets with years of (often manually) labeled data, and/or a smaller number of categories. By these standards, with 6,284 valid codes and only two years of labeled data, an automatic classification system for NAPCS should be extremely challenging to build. Despite these challenges, SINCT was able to take advantage of advances in neural networks and natural language processing to build a low-latency and highly accurate NAPCS classification tool that proved highly effective in production.

As data were collected during the 2022 Economic Census, we expanded SINCT's training data by periodically retraining the model with new respondent-provided searches and selected NAPCS codes. In many cases SINCT users found a NAPCS code they were looking for and we added these searches and NAPCS selections to the training data as a form of implicit feedback. Additionally, there were many NAPCS codes that appeared infrequently in our initial training data, so we prioritized reviewing and adding searches that might be related to these NAPCS codes to SINCT's training data.

Currently, SINCT is only in use for the Economic Census, but we hope that as NAPCS questions are added to the recently launched Annual Integrated Economic Survey, SINCT can be expanded to assist respondents with this new instrument. In addition, SINCT's approach to treating automatic classification as an information retrieval task that works directly within a questionnaire web page could be extended to other classification systems and surveys. As government statistical agencies continue to build their survey infrastructure, we hope that the challenges raised by Roberson (2021) will be addressed and that future iterations of SINCT can continue to grow with advances in neural networks.

Supplementary Material

SINCT code.

Acknowledgement

Special thanks to Taylor Hanson and Frederick King for their assistance conducting latency testing on the test server, and to Emily Wiley, Daniel Whitehead, Taylor Wilson, and Ian Thomas of the U.S. Census Bureau, and to two anonymous reviewers for reviewing drafts of this paper and providing helpful comments.

References

Büttcher S, Clarke C, Cormack G (2016). *Information Retrieval: Implementing and Evaluating Search Engines*. The MIT Press, Cambridge, MA.

Chen B, Creecy R, Appel M (1993). Error control of automated industry and occupation coding. *Journal of Official Statistics*, 9(4): 729–745.

Devlin J, Chang M, Lee K, Toutanova K (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics* (J Burstein, C Doran, T Solorio, eds.), 4171–4186. Association for Computational Linguistics, Minneapolis, MN.

Dumbacher B, Whitehead D (2024). Industry self-classification in the economic census. Accessed: July 11, 2024.

Graves A, Schmidhuber J (2005). Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural Networks*, 18: 602–610. https://doi.org/10.1016/j.neunet.2005.06.042

Hastie T, Friedman J, Tibshirani R (2011). *The Elements of Statistical Learning: Data Mining Inference and Prediction*. Springer, New York, NY.

Le Q, Mikolov T (2014). Distributed representations of sentence and documents. In: *Proceedings of the 31st International Conference on Machine Learning* (E Xing, T Jebara, eds.), volume 32, 1188–1196. Proceedings of Machine Learning Research, Beijing, China.

LeCun Y, Boser B, Denker J, Henderson D, Howard R, Hubbard W, et al. (1990). Handwritten digit recognition with a back-propagation network. In: *Advances in Neural Information Processing Systems 2, NIPS 1989*, 396–404. Morgan Kaufmann Publishers. 1989.

Measure A (2017). Deep neural networks for worker injury autocoding. Accessed: April 3, 2023.

Mikolov T, Chen K, Corrado G, Dean J (2013a). Efficient estimation of word representations in vector space. arXiv preprint: https://arxiv.org/abs/1301.3781.

Mikolov T, Sutskever I, Chen K, Corrado G, Dean J (2013b). Distributed representations of words and phrases in their compositionality. Accessed: April 3, 2023.

Mitchell T (1997). *Machine Learning* (E Munson, ed.). McGraw-Hill, New York, NY.

Moscardi C, Schultz B (2023). Using machine learning to classify products for the commodity flow survey. In: *Advances in Business Statistics, Methods and Data Collection: Introduction* (G Snijkers, M Bavdź, S Bender, J Jones, S MacFeely, J Sakshaug, K Thompson, A van Delden, eds.), 573–591. Wiley Online Library.

Office of National Statistics (2023). Automated text coding: Census 2021. Accessed: May 16, 2024.

O'Reagan (1972). Computer assigned codes from verbal responses. *Communications of the ACM*, 15: 455–459. https://doi.org/10.1145/361405.361419

Řehůřek R, Sojka P (2010). Software framework for topic modeling with large corpora. In: *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, 45–50. ELRA, Malta, Valletta.

Roberson A (2021). Applying machine learning for automatic product categorization. *Journal of Official Statistics*, 37(2): 395–410. https://doi.org/10.2478/jos-2021-0017

Roberson A, Nguyen J (2018). Comparison of machine learning algorithms to build a predictive model for classification of survey write-in responses. In: *2018 Proceedings of the Federal Committee on Statistical Methodology (FCSM) Research Conference*. FCSM, Washington, DC.

Srivastava R, Greff K, Schmidhuber J (2015). Highway networks. Access: May 16, 2024.

United States Bureau of Labor Statistics (2023). Automatic coding of injury and illness data. Accessed: April 10, 2022.

United States Census Bureau (2022). About the economic census. Accessed: April 10, 2022.

Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez A, et al. (2017). Attention is all you need, In: *Proceedings of the 30th Conference on Neural Information Processing Systems* (I Guyon, U Von Luxburg, S Bengio, H Wallach, R Fergus, S Vishwanathan, R Garnett, eds.), Curran Associates, Inc, Long Beach, CA.

Wiley E, Whitehead D (2022). Implementing interactive classification tools in the 2022 economic census. In: *2022 Proceedings of the Federal Committee on Statistical Methodology Research and Policy Conference*. FCSM, Washington, DC.