Since January 2020 Elsevier has created a COVID-19 resource centre with free information in English and Mandarin on the novel coronavirus COVID-19. The COVID-19 resource centre is hosted on Elsevier Connect, the company's public news and information website.

Elsevier hereby grants permission to make all its COVID-19-related research that is available on the COVID-19 resource centre - including this research content - immediately available in PubMed Central and other publicly funded repositories, such as the WHO COVID database with rights for unrestricted research re-use and analyses in any form or by any means with acknowledgement of the original source. These permissions are granted for free by Elsevier for as long as the COVID-19 resource centre remains active.
Harvesting Patterns from Textual Web Sources with Tolerance Rough Sets

Highlights

- We proposed a tolerance rough set learner, TPL 2.0, to harvest linguistic patterns.

- We explored concept drift phenomena and scalability of TPL 2.0.

- We extracted noun phrases from more than 900 million sentences (ClueWeb12 dataset).

- Precision values (%) @10th iteration of TPL 2.0, TPL 1.0, and CBS were 97.7, 94.5, 87.

Authors

Hoora Rezaei Moghaddam, Sheela Ramanna

Correspondence

hoora.rezaei@live.com (H.R.M.), s.ramanna@uwinnipeg.ca (S.R.)

In Brief

Construction of knowledge repositories from web corpora by harvesting linguistic patterns is of benefit for many natural language processing applications that rely on question-answering schemes. Key challenges encountered when harvesting linguistic patterns because of the iterative nature of never-ending learning are the following: number of training examples are few, relational facts might belong to more than one category depending on their context, and the emergence of concept drift where new relational facts end up being miscategorized. We propose a tolerance rough set-based learner to tackle issues of scalability and its effect on concept drift.
Harvesting Patterns from Textual Web Sources with Tolerance Rough Sets

Hoora Rezaei Moghaddam1,* and Sheela Ramanna2,3,*
1Sightline Innovation Inc., 136 Market Avenue, Unit 300, Winnipeg, MB, R3B 0P4, Canada
2Department of Applied Computer Science, University of Winnipeg, Winnipeg, Manitoba R3B 2E9, Canada
3Lead Contact
*Correspondence: hoora.rezaei@live.com (H.R.M.), s.ramanna@uwinnipeg.ca (S.R.)
https://doi.org/10.1016/j.patter.2020.100053

SUMMARY
Construction of knowledge repositories from web corpora by harvesting linguistic patterns is of benefit for many natural language-processing applications that rely on question-answering schemes. These methods require minimal or no human intervention and can recursively learn new relational facts in a fully automated and scalable manner. A key issue when mining from such a corpus is the labeling problem: data are abundant on the web but are unlabeled. Even though semi-supervised approaches are promising, they might exhibit low accuracy, because initial labeled examples of relational facts are limited in number and tend to be insufficient to properly constrain the learning process. This phenomenon is called semantic (concept) drift. We extend a recently established theoretical model for learning linguistic patterns based on tolerance rough sets to address the problem of concept drift. The choice of a tolerance rough set-based learner was motivated by the fact that the learner did not require any external constraints to constrain the learning process when compared with three benchmarked algorithms.

INTRODUCTION
Proliferation of large-scale knowledge repositories such as Wikipedia, DBpedia,1 Never Ending Learner (NELL),2 Google’s Knowledge Vault,3 and YAGO3,4,5 which store vast amount of facts about the world, has made it possible to extract interesting patterns (facts) from the aforementioned repositories. These patterns (facts) are typically referred to as categorical or relational facts. Such patterns represent entities and information about entities (e.g., James Hetfield sings for the American heavy metal band Metallica). In addition, time dimension plays an important role in discovering new facts or existing facts that can change over time. Scoping of temporal facts has been tackled in PRAVDA6 and Timely YAGO,7 and by Talukdar et al.8 An example of a temporal pattern is “Pierre Elliot Trudeau was the Prime Minister of Canada from 1968 to 1979.” Harvesting such patterns is of benefit for many natural language-processing (NLP) applications that rely on question-answering schemes. The construction of knowledge repositories is performed using semi-supervised or unsupervised machine-learning methods.
Information content for a non-temporal fact for this form of learning can be in the form of a triple (subject, predicate, category or object). This representation permits the learning of a fact (subject) as belonging to a category or semantic type using the predicate information. The predicate is an arbitrary phrase providing a context for the subject. Examples of facts are: (1) category fact, which is a unary relation (noun) that captures membership in a semantic type (represented as \texttt{Athlete Simone Biles}) and (2) relational fact, which is a binary relation (a pair of nouns) that captures semantics between entities (and membership in a semantic type) (represented as \texttt{City-In-Country (Toronto, Canada)}).

The following problems are typically encountered when harvesting linguistic patterns:

1. The number of training examples are few, i.e., facts and their known categories.
2. A relation may belong to more than one category depending on its context, leading to an inherent uncertainty of the relation to its corresponding category.
3. New relations end up being miscategorized (also known as concept drift) due to the nature of semi-supervised learning and an iterative form of never-ending learning.

To address the first problem, we use a semi-supervised learning approach in this work as shown in Figure 1 where the first learning iteration starts with a few trusted examples of facts (seeds) in a supervised manner. In every subsequent iteration, few examples from the unlabeled facts are promoted as trusted in an unsupervised manner, thus growing the knowledge repository (base) shown as output.

To address the second problem, we use co-occurrence information shown (illustrated in Figure 2) to be able to assess the degree of belongingness where a noun phrase can belong to more than one category based on its context. The rows represent the contextual patterns, the columns represent the noun phrases, and the cells represent their co-occurrence values. The third problem, which is known as concept drift (illustrated in Table 1), is handled mostly by defining external constraints such as compositional/output/or multiview-agreement constraints in Coupled Pattern Learner (CPL) or mutual exclusion in Coupled Bayesian Sets (CBS) and Fuzzy Rough Learner (FRL). Note that no constraints were used during the learning process in TPL 1.0 and TPL 2.0 for this dataset when compared with CBS and FRL.

The motivation for this work is to explore the performance of TPL 2.0 with respect to two important issues: scalability and its effect on concept drift. The benchmark algorithms given in Table 2 were used: CPL, CBS, TPL 1.0, and FRL. Performance of TPL 2.0 was evaluated with precision@30 metric using the same ontology and experimental procedures. Precision@30 is calculated as follows. In any iteration, after scoring and ranking noun phrases for a given category, the percentage of the correct instances in the set of top-30 ranked noun phrases is computed.

The scalability was tested with new larger dataset, ClueWeb12. The choice of a tolerance rough set-based learner was motivated by the fact that TPL 1.0 did not require definition of any external constraints to constrain the learning process when compared with all other benchmarked algorithms.

The contributions of this paper are: (1) extracting more than 900 million sentences from ClueWeb12 to derive our enlarged set of noun phrases; (2) preparing the co-occurrence matrix with 130,536 noun phrases and 118,648 contextual patterns, which is larger than the set used for TPL 1.0 (given in Figure 7); (3) redesigning and implementing TPL 2.0 in Python (see Figure 3); and (4) demonstrating that the TPL 2.0 algorithm produces promising results in terms of precision and handles concept drift for 20 iterations compared with the ten iterations used in all previous experiments with TPL 1.0, FRL, and CBS. The precision value at the end of the 10th iteration are as follows: TPL 2.0 (97.7%), TPL 1.0 (94.5%), CBS (87%), and FRL (89%). The precision value at the end of the 20th iteration for TPL 2.0 is 96.2%. In this paper, we refer to nouns as noun phrases.

The paper is organized as follows: In Theory, a discussion of the theoretical model used in this work is given. Experimental results using precision@30 metric at different stages of the iterative process are given, followed by a comparative analysis of TPL 2.0 with TPL 1.0, FRL, and CBS and some concluding remarks. Experimental Procedures details the data-extraction process from the web archive, and the TPL algorithm used in learning patterns is presented. This section also presents issues related to the time
complexity of the TPL algorithm. A complete trace of the TPL 2.0 algorithm with sample data can be found in Tables S1–S5.

Theory
Here, we present some background related to handling uncertainty with rough and fuzzy sets as well as basic notations of rough sets and tolerance rough sets. A more detailed theoretical discussion on tolerance and fuzzy rough sets can be found in Bahradwaj,16 Moghaddam,17 and Sengoz.18

Handling Uncertainty with Granular Methods
Rough set theory19 and Fuzzy set theory20 are core theories that constitute granular methods to handle uncertainty and vagueness. These methods use information granules where a granule is a clump of objects (points) in the universe of discourse drawn together by indistinguishability, similarity, proximity, or functionality.20 Reasoning with granular methods is particularly useful in natural language applications with uncertain class boundaries. In classical rough sets theory, a universe of objects is partitioned into indiscernible classes (i.e., granules) by means of an equivalence relation. Indiscernible classes form basic granules of knowledge about the universe. Given a concept that is determined to be vague (not precise), this theory makes it possible to express the vague concept by a pair of precise concepts termed as lower and upper approximations. A vague concept is a class (or category) that cannot be uniquely classified. The difference between the upper and the lower approximations constitutes the boundary region of the vague concept. Hence, rough set theory handles vagueness not by means of membership but by using the boundary region.

In fuzzy set theory, the relational counterpart generates soft similarity classes that permit partial overlap, even when the fuzzy relation is reflexive, symmetrical, and $T$-transitive, i.e., a so-called fuzzy $T$-equivalence relation.21 This is the property that lies at the heart of fuzzy-rough set models and lends itself to natural language applications.22

For NLP applications, classical rough set theory based on equivalence relations is considered too restrictive. In contrast, a tolerance form of rough set theory23,24,25 based on tolerance relations is more appropriate. Mathematically, tolerance relations are reflexive and symmetric but are not necessarily transitive, so the classes induced by such relations may have overlapping members. An in-depth survey of application of tolerance rough sets in text categorization can be found in Ramanna et al.26

Rough Sets: Formal Notation
Let $U$ be a finite, non-empty universe of objects and let $R \subseteq U \times U$ denote a binary relation on the universe $U$. $R$ is called an indiscernibility relation and for rough sets, it has to be an equivalence relation. The pair

\[(U, R) = \mathcal{A}\]

constitutes an approximation space.27 Assume we have $X \subseteq U$ as a target concept in this universe. The task, then, is to create an approximated representation for $X$ in $U$ with the help of $R$. Let $[x]_R$ denote the indiscernibility class of $x$, i.e., $y \in [x]_R \iff (x, y) \in R$. Then every equivalence class forms a granule or partition which, as the name implies, contains objects that are indiscernible for this approximation space $\mathcal{A}$. In other words, every single item in a granule is considered identical and inseparable. Eventually, these granules are approximated by the following two operators:

- **Lower approximation.** Informally, these are the objects that certainly belong to $X$ with respect to $\mathcal{A}$, shown as orange squares in Figure 4.

\[\mathcal{L}_\mathcal{A}(X) = \{x \in U : [x]_R \subseteq X\}. \quad \text{(Equation 1)}\]
Table 2. Benchmark Algorithms Used in This Work

| CPL               | CBS               | TPL 1.0           | FRL               |
|-------------------|-------------------|-------------------|-------------------|
| semi-supervised algorithm | semi-supervised algorithm | semi-supervised algorithm | semi-supervised algorithm |
| core component of NELL | based on Bayesian sets | based on rough sets | based on fuzzy-rough sets |
| corpus of 200 million web pages | ClueWeb09 corpus (subset) | ClueWeb09 corpus (subset) | ClueWeb09 corpus (subset) |
| outperforms CPL after 10 iterations | outperforms CBS after 10 iterations | comparable with TPL 1.0 and CBS after 10 iterations |

- **Upper approximation.** Informally, these are the objects which may belong to X with respect to A, shown as combined orange and green squares in Figure 4.

\[ U_A(X) = \{ x \in U : [x]_A \cap X \neq \emptyset \} \quad \text{(Equation 2)} \]

These two approximations will lead to the definition of the following two regions:

- **Boundary region.** These are the objects occurring in the upper approximation but not in lower approximation of X, shown as green squares in Figure 4.

\[ B_A(X) = U_A(X) - L_A(X) \quad \text{(Equation 3)} \]

- **Negative region.** These are the objects that certainly do not belong to X, shown as gray squares in Figure 4.

\[ U - U_A(X) \quad \text{(Equation 4)} \]

It should be noted that each granule can contain an arbitrary number of objects or may be empty. They are depicted as squares only for the sake of illustration. In this paper, the universe \( U \) will consist of such items as linguistic entities (relational facts) and contextual patterns. The target concept \( X \) consists of categories (semantic type).

**Tolerance Rough Sets: Formal Notation**

A tolerance relation \( I \subseteq U \times U \) can be any binary relation that is reflexive and symmetric. It will be used as an indiscernibility relation for a tolerance form of rough sets. Because it is not transitive, indiscernibility classes induced by such relations can overlap.

An indiscernibility class induced by a tolerance relation is called a tolerance class:

\[ l(x) = \{ y \in U | (x, y) \in I \} \]

In this work we use the tolerance approximation space model proposed by Skowron and Spepaniuk. A tolerance approximation space is denoted by

\[ T = (U, I, r, P) \quad \text{(Equation 5)} \]

where:

- **Universe** \( U \) is the universe of objects.
- **Uncertainty function** \( l : U \to [0, 1] \) given that \( P(U) \) is the power set of \( U \). It defines the tolerance class of an object. It also implicitly defines the tolerance relation \( I \) such that \( xIy \Leftrightarrow r(y) \leq r(x) \). It can be any relation that is reflexive and symmetric. It essentially defines a neighborhood.
- **Vague inclusion function** \( r : P(U) \times P(U) \to [0, 1] \) measures the degree of inclusion between two sets. It can be any function that is monotone with respect to the second argument: \( Y \subseteq Z \Rightarrow r(Y, Z) \leq r(Z, Z) \) for \( X, Y, Z \subseteq U \).
- **Structurality function** \( P : l(U) \to [0, 1] \), where \( l(U) = \{ l(x) : x \in U \} \) allows additional binary conditions to be defined over the tolerance classes.

The lower and upper approximations of set \( X \) can then be defined as

\[ L_T(X) = \{ x \in U : P(l(x)) = 1 \land r(l(x), X) = 1 \} \quad \text{(Equation 6)} \]

\[ U_T(X) = \{ x \in U : P(l(x)) = 1 \land r(l(x), X) > 0 \} \quad \text{(Equation 7)} \]

**Tolerance Rough Sets Adapted to NLP**

We can now define a tolerance approximation space categorizing noun phrases as \( K = (CP, NP, I, \lambda, \eta) \). In this work, the universes of entities are:

- **NP** = \( \{ np_1, np_2, ... np_v \} \) represents the universe of noun phrases. Each np can be a word or a group of words that functions in a sentence as subject or object or
prepositional object. Examples of noun phrases are: “rationalism,” “light intensity,” “Freedom of Information Act,” ….  

- **CP** = {cp₁, cp₂, …, cpₙ} represents the universe of contextual patterns. Each cp belongs to some specific lexical category. Examples of contextual patterns are: “is available,” “quality comes from,” “celebrities such as,” “cooperative agreement with,” ….  

Our approximation space framework for categorical information classification includes two mapping functions that associate every noun phrase to its contextual patterns and contextual patterns to nouns:  

- C : NP → ℙ(CP) maps each noun phrase to its corresponding set of co-occurrence categorical context as C(npi) = {cpᵢ : fᵢ(cpᵢ, npi) > 0}, where fᵢ(cpᵢ, npi) occurs ϕ times within context cpᵢ.  

- N : CP → ℙ(NP) maps each contextual category to its set of co-occurring noun phrases where N(cpᵢ) = {npi : fᵢ(cpᵢ, npi) > 0}.  

The parameterized uncertainty function I = Iₜ is defined as  

\[ Iₜ(cpᵢ) = \{ cpᵢ : \lambda(N(cpᵢ), N(cpᵢ)) \geq \theta \}. \] (Equation 8)  

Here, Iₜ(cpᵢ) gives the tolerance class for each contextual pattern cpᵢ and θ is the threshold value. By tuning the threshold value, one can control the relatedness of the conceptual patterns of noun phrases in the tolerance class. In our model we have used λ as our overlap index, which is based on the Sorensen-Dice index:\(^{29}\)  

\[ \lambda(M, N) = \frac{2|M \cap N|}{|M| + |N|} \] (Equation 9)  

where M and N are defined as N(cpᵢ) and N(cpᵢ) for our case. The vague inclusion function is defined as  

\[ \eta(S₁, S₂) = \frac{|S₁ \cap S₂|}{|S₁|}, \] (Equation 10)  

which evaluates the degree of inclusion when η : ℙ(CP) × ℙ(CP) → [0, 1]. Based on the framework of K, one can calculate the lower and upper approximations as  

\[ \mathcal{L}_K(npi) = \{ cpᵢ \in CP : \eta(Iₜ(cpᵢ), C(npi)) = 1 \}. \] (Equation 11)  

\[ \mathcal{U}_K(npi) = \{ cpᵢ \in CP : \eta(Iₜ(cpᵢ), C(npi)) > 0 \}. \] (Equation 12)  

which basically gives us a set of related contextual patterns. In this case, \( \mathcal{L}_K(npi) \) will provide us with a subset from CP that is “strongly” related contexts to its co-occurring noun phrase while \( \mathcal{U}_K(npi) \) will provide us with a subset of CP that is “weakly” related to its co-occurring noun phrase.  

Figure 5 illustrates the partitioning of the universe into four different zones of a candidate noun phrase based on the approximation framework. Each zone contributes to a specific weight that determines the promotion status of a candidate noun phrase. As a result, every promoted noun phrase npᵢ is associated to three sets: C(npi), \( \mathcal{U}_K(npi) \), and \( \mathcal{L}_K(npi) \). These sets are used to calculate a score for a candidate instance npᵢ, by the trusted instance npᵢ:  

\[ \text{score}(npᵢ, npᵢ) = \lambda(C(npi), C(npi)) \alpha + \lambda(\mathcal{L}_K(npi), C(npi)) \beta + \lambda(\mathcal{U}_K(npi), C(npi)) \gamma. \] (Equation 13)  

Parameters α, β, γ are the contributing factors for the scoring components and they can be configured for the application domain. A trusted instance npᵢ has the universe of contexts partitioned by its sets \( \mathcal{L}_A(npi) \), C(npi), and \( \mathcal{U}_A(npi) \) into four zones of recognition. For a candidate npᵢ, each zone will represent a different degree of similarity. When calculating the microscore, the candidate’s contexts falling in zone 1 (lower approximation) will be covered by all three sets and will thus make a high contribution to its score. Contexts in zone 2 will be covered by C(npi) and \( \mathcal{U}_A(npi) \) so they will make medium contribution. Zone 3 contexts will only be covered by \( \mathcal{U}_A(npi) \) and will make low contribution. Contexts in zone 4 will not contribute at all, since they suggest no resemblance between npᵢ and npᵢ. These approximation operators will assess whether these linguistic patterns can be categorized as “certain,” “uncertain,” “negative,” or in “boundary” zone of a semantic type.  

RESULTS  

Throughout our experiments, we used the same process as CBS, TPL 1.0, and FRL. We used the same 11 categories in our ontology: Company, Disease, KitchenItem, Person, PhysicalTerm, Plant, Profession, Sociopolitics, Website, Vegetable, and Sport. Each category was initialized with 5–8 seed instances and we let TPL 2.0 run for 20 iterations. For every category, we had the top-5 new noun phrases promoted as “trusted” per iteration, which were then used as seeds in the subsequent iterations. We heuristically set the tolerance threshold to θ = 50%, since it led us to the most semantically accurate tolerance classes as in TPL 1.0.\(^{30}\) Precision@30 metric was used for evaluating TPL 2.0 performance, similar to CBS, TPL 1.0, and FRL. In each iteration, Precision@N is calculated as follows:  

\[ \text{Precision}@N = \frac{\text{number of correct examples}}{N \text{ top } - \text{ranked examples}}. \] (Equation 15)
As the dataset is not labeled, the correctness of an instance was judged manually. We report results with two sets of parameters. In Table 3, the parameters values are $\alpha = 0.5$, $\beta = 0.75$, $\gamma = 0.25$. In Table 4, the parameters were set to the same values that were used in TPL 1.0. The results in Table 4 clearly demonstrate the precision value drop after iteration 7. This shows that for this parameter setting concept drift occurs. It should be noted that the parameters have to be tuned to the specific dataset.

**DISCUSSION**

For a vast majority of the categories, TPL 2.0 successfully achieved high precision over ten iterations without showing any sign of semantic drifting. Table 3 quantitatively illustrates the performance of TPL 2.0 using Precision@30 for each category per iteration. TPL 2.0 was able to learn the following categories with 100% precision: Person, Company, Disease, Profession, and Sport. With categories KitchenItem and Plant, only two instances of candidate noun phrases were promoted incorrectly. Category Sociopolitics was also difficult to learn. However, category Vegetable was the least accurate due to the presence of ambiguous contextual patterns associated with it. The ambiguity was due to the fact that both poultry and fruits were classified as Vegetables. It is noteworthy that for this category, the precision value of 73% is higher than CBS, TPL 1.0, and FRL.

For some categories (see Table 5), it is not straightforward to infer boundaries. As a result we noticed some semantic drifting. To demonstrate this point, consider the category Sociopolitics. This is a vague category for which no apparent boundary exists (a table showing detailed results can be found in Moghaddam17). On the other hand, one can observe that in iteration 20, some noun phrases were getting not accepted as belonging to certain categories. For instance, for the category Website, even though Skype is also a website, TPL ignores the Website category in the initial iterations and subsequently categorizes it as Company,

### Table 3. TPL 2.0 Results with Precision@30 for All Categories, by Iteration (%) with $\alpha = 0.5$, $\beta = 0.75$, $\gamma = 0.25$

| Category     | Iteration 1 | Iteration 3 | Iteration 5 | Iteration 7 | Iteration 10 | Iteration 13 | Iteration 15 | Iteration 17 | Iteration 20 |
|--------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|
| Company      | 100         | 100         | 100         | 100         | 100         | 100         | 100         | 100         | 100         |
| Disease      | 100         | 100         | 100         | 100         | 100         | 100         | 100         | 100         | 100         |
| KitchenItem  | 97          | 97          | 97          | 100         | 97          | 100         | 100         | 100         | 100         |
| Person       | 100         | 100         | 100         | 100         | 100         | 100         | 100         | 100         | 100         |
| PhysicsTerm  | 100         | 100         | 97          | 94          | 97          | 97          | 94          | 97          | 100         |
| Plant        | 100         | 100         | 94          | 100         | 97          | 84          | 94          | 94          | 94          |
| Profession   | 100         | 100         | 100         | 100         | 100         | 100         | 100         | 100         | 100         |
| Sociopolitics| 94          | 97          | 94          | 97          | 97          | 90          | 94          | 87          | 87          |
| Website      | 97          | 87          | 97          | 97          | 97          | 100         | 90          | 97          | 100         |
| Vegetable    | 77          | 84          | 74          | 84          | 90          | 84          | 74          | 74          | 77          |
| Sport        | 100         | 100         | 100         | 100         | 100         | 100         | 100         | 100         | 100         |
| Average      | 95.4        | 96.8        | 95.7        | 96.8        | 97.7        | 95.9        | 95.1        | 95.3        | 96.2        |
which is still a correct assumption as Skype Inc. was purchased by Microsoft in 2011. Fortunately, most categories including Profession, Plant, Company, Person, Sport, and Disease had clear boundaries. One of the most important factors that affected the performance of the TPL 2.0 were categories that are subsets of each other or are relatively smaller subsets of another big category. PhysicsTerm is a good example. Here, there were noun phrases spin or atom with co-occurrence contextual patterns that appeared with other science subject terms related to chemistry (e.g., double bond) or mathematics (e.g., conic section). These were discovered in the 20th iteration. The other example is the Vegetable category. One solution is to combine meat (e.g., mutton), poultry (e.g., salami), and fruits (e.g., kumquat) into a single category name Comestible.

Concluding Remarks
In this paper, to test the scalability of tolerance rough set-based pattern learner we have tackled the problem of extracting information from a big noisy dataset of crawled web pages. As data preparation is the most challenging and time-consuming component at the core of the TPL 2.0 system, we conducted extensive experiments to propose an efficient way for handling data preparation. We proposed a semi-automated data-preparation procedure to extract more than 900 million sentences based on the rules of English grammar, crafting our set of noun phrases as well as contextual patterns. We have also shown the effect of two different sets of parameter values as compared with TPL 1.0 for this specific dataset. We have demonstrated that the proposed TPL 2.0 algorithm shows promise with Precision@30 metric in handling concept drift for twice the number of iterations as compared with the previous experiments with our benchmark algorithms. Future work will include categorizing facts from microblogging sites such as twitter data.

Limitations of this Study
This study was limited in terms of the size of the dataset and number of iterations. This study was meant to answer the following question: Can TPL 2.0 learn to populate many different categories and nouns for dozens of iterations of learning and maintain high precision? To truly assess the robustness of TPL 2.0 over several iterations, a much larger dataset must be prepared. Another limitation was a lack of

Table 4. TPL 2.0 Results with Precision@30 for All Categories, by Iteration (%) with $\alpha = 0.5$, $\beta = 0.25$, $\gamma = 0.25$

| Iteration | Company | Disease | KitchenItem | Person | PhysicsTerm | Plant | Profession | Sociopolitics | Website | Vegetable | Sport | Average |
|-----------|---------|---------|-------------|--------|-------------|-------|------------|--------------|---------|-----------|-------|---------|
| 1         | 100     | 100     | 100         | 100    | 100         | 100   | 100        | 100          | 87      | 77        | 100   | 96.7    |
| 3         | 100     | 100     | 100         | 100    | 100         | 100   | 100        | 100          | 90      | 90        | 100   | 97.6    |
| 5         | 100     | 100     | 100         | 100    | 100         | 100   | 100        | 100          | 90      | 94        | 100   | 97.5    |
| 7         | 100     | 100     | 97          | 100    | 97          | 100   | 100        | 97           | 97      | 87        | 100   | 97.3    |
| 10        | 100     | 100     | 97          | 100    | 97          | 100   | 100        | 97           | 97      | 94        | 100   | 94.27   |
| 13        | 100     | 100     | 94          | 100    | 97          | 100   | 100        | 97           | 84      | 94        | 100   | 94.63   |
| 15        | 100     | 100     | 100         | 100    | 100         | 100   | 100        | 94           | 94      | 94        | 100   | 93.72   |
| 17        | 100     | 100     | 100         | 100    | 100         | 100   | 100        | 93           | 97      | 94        | 100   | 93.18   |
| 20        | 100     | 100     | 100         | 100    | 100         | 100   | 100        | 100          | 100     | 100       | 100   | 93.81   |

Table 5. TPL 2.0 and TPL 1.0 and FRL and CBS Results with Precision@30 per Category

| Categories    | Iteration 5 |       |       |       | Iteration 10 |       |       |       |
|---------------|-------------|-------|-------|-------|--------------|-------|-------|-------|
|               | TPL 2.0     | TPL 1.0 | CBS   | FRL  | TPL 2.0      | TPL 1.0 | CBS   | FRL  |
| Company       | 100         | 100    | 100   | 100   | 100          | 100    | 100   | 100   |
| Disease       | 100         | 100    | 100   | 100   | 100          | 100    | 100   | 100   |
| KitchenItem   | 97          | 100    | 94    | 97    | 100          | 97     | 94    | 94    |
| Person        | 100         | 100    | 100   | 100   | 100          | 100    | 100   | 100   |
| PhysicsTerm   | 97          | 93     | 100   | 67    | 97           | 97     | 90    | 77    |
| Plant         | 94          | 100    | 100   | 77    | 97           | 97     | 100   | 100   |
| Profession    | 100         | 100    | 100   | 100   | 100          | 100    | 87    | 100   |
| Sociopolitics | 94          | 100    | 47    | 93    | 97           | 100    | 34    | 87    |
| Sport         | 100         | 97     | 97    | 100   | 100          | 100    | 100   | 100   |
| Website       | 97          | 90     | 94    | 97    | 97           | 97     | 90    | 93    |
| Vegetable     | 74          | 93     | 83    | 83    | 90           | 63     | 48    | 47    |
| Average       | 95.7        | 97.5   | 92    | 92    | 97.7         | 94.5   | 87    | 89    |

CBS results are as reported in Verma and Hruschka. FRL results are as reported in Bharadwaj and Ramanna. TPL 1.0 results are as reported in Sengoz and Ramanna.
sensitivity analysis necessary to study the setting of the \( \alpha \), \( \beta \), and \( \gamma \) parameters. Given the current COVID-19 lockdown situation restricting access to our laboratory, this experiment could not be carried out. The claim made here is that TPL 2.0 shows promising results. However, it is also likely that the TPL learning process may need to be constrained as the number of iterations increases.

**EXPERIMENTAL PROCEDURES**

**Resource Availability**

**Lead Contact**

Sheela Ramanna is the Lead Contact for this paper. s.ramanna@uwinnipeg.ca.

**Materials Availability**

There were no materials generated by this study.

**Data and Code Availability**

TPL 2.0 code can be downloaded from https://drive.google.com/drive/folders/1cibEecpjmxNwDNOo9Rnhb86SiaZn1w?usp=sharing (Ramanna, Sheela; Moghaddam, Hoora Rezaei (2020), “TPL2.0Code”, Mendeley Data, V1, DOI: 10.17632/jydpbt3z4d.1). The authors are unable to make the dataset public due to the agreement with Carnegie Mellon University regarding the use of ClueWeb12 dataset.

The original data source is a dataset created by Lemur Project ClueWeb12,3 which is successor to the ClueWeb09 dataset. ClueWeb12 is a collection of about 733,019,372 English web pages (approximately 6TB of compressed data) in Web Archive format (https://www.iso.org/standard/44717.html).

**Extractor Module Details**

The categorical noun phrase extractor module of TPL 2.0 was implemented in Python, since the input data are in the form of a sparse (noun phrase-contextual pattern co-occurrence) matrix. A multi-thread programming on Ubuntu 16.04 LTS machine with 2 Intel Xeon CPU 12-core (2.5GHz) 96GB memory 900GB 10K was used in this work. TPL 1.0 noun phrase repository was used as a starting repository for TPL 2.0. For each noun phrase, approximately 10,000 and 48,000 sentences were extracted by retaining the core of the statement as shown below.

**Algorithm**

Algorithm 3.2 gives the pseudo-code for TPL 2.0. This algorithm was initialized with seed instances (initial trusted nouns) for the first iteration of training and uses a score-based ranking with a scoring mechanism similar to TPL 1.0.28

---

**Figure 6. Sample Parts-of-Speech Tags for Extracting Contextual Patterns**
Algorithm 1: Tolerant Pattern Learner TPL 2.0 for Categories

Input: An ontology \( O \) defining categories and a small set of seed examples; a large corpus \( U \).

Output: Trusted noun phrases for each category \( c_i \)

for \( \text{iter} = 1 \rightarrow \infty \) do
    for each category such as \( c_d \) do
        for each new trusted noun phrase \( np_i \) of \( c_d \) do
            Calculate the approximations \( L_{K_i}(np_i) \) and \( L_{K_i}(np_i) \) and \( C(np_i) \);
            For each candidate noun phrase \( np_j \) do
                Calculate score \( (np_i, np_j) \) (Eq.13);
            end
        end
    end
    for each candidate noun phrase \( np_j \) do
        \( TS_{c_i}(np_j) = \sum_{np_i \in c_d} \text{score}(np_i, np_j) \) (Eq.14);
    end
    Rank instances by \( TS_{c_i}/|c_i| \);
    if \( TS_{c_i} \geq (\text{Avg}(TS_{c_i}))_{\text{iter}-1} \) then
        Promote top instances as trusted;
    else
        continue
    end
end
end
end
end

Each category is identified by its trusted nouns. Consequently, each trusted noun acts as a "proxy" for its own category. Ranking of candidate noun phrases is performed based on their similarity to the trusted nouns. For a given category \( c_i \), we can determine a total score \( TS \) for the candidate \( np_j \) in category \( c_i \) as

\[
TS_{c_i}(np_j) = \sum_{np_i \in c_i} \text{score}(np_i, np_j),
\]

(Equation 14)

where \( TS_{c_i}(np_j) \) is the total score for the \( j \)th noun phrase \( np_j \) for the \( i \)th category \( c_i \). \( TN_{c_i} \) also represents the set of trusted nouns for the \( i \)th category \( c_i \). After calculating the total score for every candidate for each \( c_i \), we rank the candidates by their \( TS \)s (normalized by the number of trusted instances of each \( c_i \)). Eventually, we promote the top candidates as new trusted nouns, which have the score higher or equal to the average of the top ten candidates of the previous round.

TPL 2.0 learns in an iterative manner and in every iteration, it learns new trusted instances and their categories. Since the first iteration is using user-labeled nouns as the initial seeds, this step forms the supervised learning stage of the algorithm, while the next iterations use the automatically extracted instances and therefore form the unsupervised learning stages. A complete trace of TPL 2.0 is given in the Supplemental Information.

Complexity and Scalability Issues

The time complexity of the TPL 2.0 algorithm is affected by the following parameters: number of categories, number of iterations, size of the noun phrases set \( |NP| \), size of the contextual pattern set \( |CP| \), and number of trusted noun phrases promoted for each category. For TPL 2.0, the time-consuming part of the decision making is to calculate the upper and lower approximations for each trusted noun. Since calculating tolerance classes for the co-occurring contextual patterns with trusted noun phrases is the main resource for calculating the lower and upper approximation, one can conclude that having more trusted noun phrases will result in more overhead. In this work, the number of iterations was set to 20, number of categories was set to 11, and number of trusted noun phrases to be promoted for each category was set to 5 per each round as discussed. Therefore, the run time will depend on \( |NP| \) and \( |CP| \). Technically, the sparsity of the co-occurrence matrix shown in Figure 2 will affect the complexity of TPL 2.0. Eventually, time complexity for TPL 2.0 will be bounded by \( O(|CP|) \). This is the same as TPL 1.0.

**SUPPLEMENTAL INFORMATION**

Supplemental Information can be found online at https://doi.org/10.1016/j.patter.2020.100053.

**ACKNOWLEDGMENT**

Special thanks to Cenker Sengoz and Aditya Bharadwaj for discussions regarding TPL 1.0 and FRL, and to Vu Pham Van for preparing an executable version of TPL 2.0. We are very grateful to Prof. Estevam R. Hruschka Jr. for the CBS (CluWeb09-NELL) dataset. S.R.’s research is supported by the Natural Sciences and Engineering Research Council of Canada discovery grant 194376.
AUTHOR CONTRIBUTIONS

S.R. designed the materials and methods and prepared the paper. H.R.M. developed the code, performed the experiments, and contributed to the writing of the paper.

DECLARATION OF INTERESTS

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

REFERENCES

1. Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P.N., Hellmann, S., Morsey, M., van Kliief, P., Auer, S., and Bizer, C. (2014). DBpedia—a large-scale, multilingual knowledge base extracted from Wikipedia. Semantic Web J. https://doi.org/10.3233/SW-140134.
2. Mitchell, T., Cohen, W., Huang, B., Hruschka, E., and Mitchell, T. (2012). Coupled temporal classification of relational facts. In Proceedings of the Third ACM International Conference on Web Search and Data Mining, pp. 101–110.
3. Bharadwaj, A., and Ramanna, S. (2019). Categorizing relational facts from the web with fuzzy rough sets. Knowledge Inf. Syst. 61, 1695–1713.
4. Callan, J. (2012). The Lemur Project and its ClueWeb12 dataset, Invited talk at the SIGIR 2012 Workshop on Open-Source Information Retrieval http://opensearchlab.otago.ac.nz/SIGIR12-OSIR-callan.pdf.
5. Bharadwaj, A. (2017). Fuzzy Rough Set Based Unstructured Text Categorization. Master’s Thesis (University of Winnipeg), Supervisor, S. Ramanna.
6. Rezaei Moghaddam, H. (2019). Exploring Scalability and Concept Drift Issues in Learning Categorical Facts with Tolerance Rough Sets. Master’s Thesis (University of Winnipeg), Supervisor, S. Ramanna.
7. Sengoz, C. (2014). A Granular-Based Approach for Semisupervised Web Information Labeling. Master’s Thesis (University of Winnipeg), Supervisor, S. Ramanna.
8. Mahdisoltani, F., Biega, J., and Suchanek, F.M. (2015). YAGO3: a knowledge base from multilingual Wikipedias. In Proceedings of the 7th Biennial Conference on Innovative Data Systems Research (CIDR 2015) http://cidrdb.org/cidr2015/Papers/CIDR15_Paper1.pdf.
9. Suchanek, F., Kasneci, G., and Weikum, G. (2007). Yago: a core of semantic knowledge. In Proceedings of the 16th International World Wide Web Conference (WWW’07) (ACM Press), https://doi.org/10.1145/ 1242572.1242667.
10. Wang, Y., Yang, B., Qu, L., Spaniol, M., and Weikum, G. (2011). Harvesting facts from textual web sources by constrained label propagation. In Proceedings of the 20th Conference on Information and Knowledge Management (ACM Press), pp. 837–846.
11. Wang, Y., Yang, B., Qu, L., Spaniol, M., and Weikum, G. (2010). Timely YAGO: harvesting, querying, and visualizing temporal knowledge from Wikipedia. In Proceedings of the 13th International Conference on Extending Database Technology (ACM Press), pp. 697–700.
12. Talukdar, P.P., Wijaya, D., and Mitchell, T. (2012). Coupled temporal scoping of relational facts. In Proceedings of the Fifth ACM International Conference on Web Search and Data Mining (WSDM) (ACM), pp. 73–82.
13. Carlsson, A., Betteridge, J., Wang, R.C., Hruschka, E.R., Jr., and Mitchell, T.M. (2010). Coupled semi-supervised learning for information extraction. In Proceedings of the Third ACM International Conference on Web Search and Data Mining, pp. 101–110.
14. Bharadwaj, A., and Ramanna, S. (2019). Categorizing relational facts from the web with fuzzy rough sets. Knowledge Inf. Syst. 61, 1695–1713.
15. Callan, J. (2012). The Lemur Project and its ClueWeb12 dataset, Invited talk at the SIGIR 2012 Workshop on Open-Source Information Retrieval http://opensearchlab.otago.ac.nz/SIGIR12-OSIR-callan.pdf.
16. Bharadwaj, A. (2017). Fuzzy Rough Set Based Unstructured Text Categorization. Master’s Thesis (University of Winnipeg), Supervisor, S. Ramanna.
17. Rezaei Moghaddam, H. (2019). Exploring Scalability and Concept Drift Issues in Learning Categorical Facts with Tolerance Rough Sets. Master’s Thesis (University of Winnipeg), Supervisor, S. Ramanna.
18. Sengoz, C. (2014). A Granular-Based Approach for Semisupervised Web Information Labeling. Master’s Thesis (University of Winnipeg), Supervisor, S. Ramanna.
19. Zdzisław Pawlak. (1982). Rough sets. Int. J. Comput. Inf. Sci. 11, 341–356.
20. Zadeh, L.A. (1997). Towards a theory of fuzzy information granulation and its centrality in human reasoning and fuzzy logic. Fuzzy Sets Syst. 177 (19), 111–127.
21. De Cock, M., and Cornelis, C. (2005). Fuzzy rough set based web query expansion. In Proceedings of Rough Sets and Soft Computing in Intelligent Agent and Web Technology, pp. 9–16.
22. Dubois, D., and Prade, H. (1990). Rough fuzzy sets and fuzzy rough sets. Int. J. Gen. Syst. 17, 191–209.
23. Marcus, S. (1994). Tolerance rough sets, czech topologies, learning processes. Bull. Polish Acad. Sci. Tech. Sci. 42, 471–487.
24. Nieminen, J. (1988). Rough tolerance equality and tolerance black boxes. Fundamenta Informaticae 11, 289–296.
25. Polkowski, L., Skowron, A., and Zytkow, J. (1994). Tolerance based rough sets. In Soft Computing: Rough Sets, Fuzzy Logic, Neural Networks, Uncertainty Management, Knowledge Discovery, T.Y. Lin and M. Wildberger, eds. (Simulation Councils Inc), pp. 55–68.
26. Ramanna, S., Peters, J.F., and Sengoz, C. (2017). Application of tolerance rough sets in structured and unstructured text categorization: a survey. In Thriving Rough Sets, G. Wang, A. Skowron, Y. Yao, D. Slezak, and P. Polkowski, eds. (Springer), pp. 119–138.
27. Skowron, A., and Stepaniuk, J. (1996). Tolerance approximation spaces. Fundam. Inf. 27, 245–253.
28. Sengoz, C., and Ramanna, S. (2014). A semi-supervised learning algorithm for web information extraction with tolerance rough sets. In Active Media Technology 10th International Conference, AMT 2014, D. Slezak, G. Schaefer, S.T. Vuong, and Y.S. Kim, eds. (Springer), pp. 1–10.
29. Sorensen, Th.A. (1948). A method of establishing groups of equal amplitude in plant sociology based on similarity of species content and its application to analyses of the vegetation on Danish commons. Biol. Skar. 5, 1–34.
30. Sengoz, C., and Ramanna, S. (2015). Learning relational facts from the web: a tolerance rough set approach. Pattern Recognition Lett. 67, 130–137.
31. Bird, S., Klein, E., and Loper, E. (2009). Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit (O’Reilly Media, Inc.).
32. Santorini, B. Part-of-speech tagging guidelines for the Penn Treebank Project, 1990. Technical Report, CIS 7-1-1990.
33. Toutanova, K., and Manning, C.D. (2000). Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. In Proceedings of the 2000 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (Association for Computational Linguistics), pp. 63–70.