Osprey: Weak Supervision of Imbalanced Extraction Problems without Code

Eran Bringer, Abraham Israeli
{eran.bringer,abraham.israeli}@intel.com
Intel Corporation
Haifa, Israel

Alex Ratner, Christopher Ré
{ajratner,chrismre}@cs.stanford.edu
Stanford University
Palo Alto, CA, USA

ABSTRACT

Supervised methods are commonly used for machine-learning based applications but require expensive labeled dataset creation and maintenance. Increasingly, practitioners employ weak supervision approaches, where training labels are programmatically generated in higher-level but noisier ways. However, these approaches require domain experts with programming skills. Additionally, highly imbalanced data is often a significant practical challenge for these approaches. In this work, we propose `Osprey`, a weak-supervision system suited for highly-imbalanced data, built on top of the Snorkel framework. In order to support non-coders, the programmatic labeling is decoupled into a code layer and a configuration one. This decoupling enables a rapid development of end-to-end systems by encoding the business logic into the configuration layer. We apply the resulting system on highly-imbalanced (0.05% positive) social-media data using a synthetic data rebalancing and augmentation approach, and a novel technique of ensembling a generative model over the legacy rules with a learned discriminative model. We demonstrate how an existing rule-based model can be transformed easily into a weakly-supervised one. For 3 relation extraction applications based on real-world deployments at Intel, we show that with a fraction of the cost, we achieve gains of 18.5 precision points and 28.5 coverage points over prior traditionally supervised and rules-based approaches.

KEYWORDS

weak supervision. machine learning democratization, end-to-end systems, relation extraction

DEEM, June 2019, Amsterdam, NL

ACM ISBN 978-1-4503-9999-9/18/06…$15.00
https://doi.org/10.1145/1122445.1122456

ACM Reference Format:
Eran Bringer, Abraham Israeli and Alex Ratner, Christopher Ré. 2019. `Osprey`: Weak Supervision of Imbalanced Extraction Problems without Code. In *Proceedings of ACM Conference (DEEM).* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

In recent years, modern machine learning (*ML*) models have become increasingly powerful but also complex, achieving new state-of-the-art results on a range of traditionally challenging tasks, but requiring massive hand-labeled training sets to do so [19]. However, while some labeled data sets are available for more generic or benchmark problems, this is not the case for the domain-specific, dynamically-changing problems of real-world users. For example, many labeled data sets are publicly available for the generic task of sentiment analysis—but none for extracting custom-defined ”business partnership” relations from text feeds. In response, many ML developers have increasingly turned to *weak supervision* methods, in which a larger volume of more cheaply-generated, noisier training labels is used in lieu of a smaller hand-labeled set. Especially given the increasing commoditization of standard ML model architectures, the supervision strategy used is increasingly the key differentiator for end model performance, and recently has been the key technique in state-of-the-art results [6, 13]. Prior work in weak supervision has focused on the setting of independent crowd workers [7, 9], custom-tailored and hand-tuned *distant supervision* strategies in the natural language processing domain [14, 21], knowledge-bases[12, 14], handling generic label noise or misspecification [3, 15]. Recent work has focused on building end-to-end systems allowing non-experts to create and manage multiple sources of weak supervision that may have diverse accuracies, coverages, and correlations [1, 18].

The Snorkel framework for weakly-supervised ML [17] allows users to generically specify multiple sources of weak supervision that vary in accuracy, coverage, and that may be arbitrarily correlated. Snorkel's pipeline follows three main stages: first, users write *labeling functions* (LFs), which are simply black-box functions that take in unlabeled data points and output a label or abstain, and can be used to express a wide variety of weak supervision strategies; next,

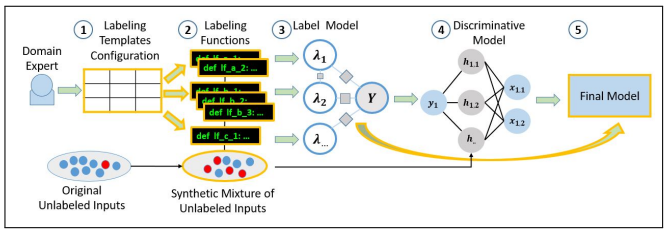

Figure 1: In the Osprey pipeline, rather than manually or programmatically labeling training data, domain experts configure the labeling templates through a simple tabular interface (1) from which groups of labeling-function variants are generated by the LF Generator (2). This weak supervision is applied to a synthetically-balanced dataset and automatically denoised by a generative model (3), producing labels for training a discriminative model such as deep neural network (4). The generative and discriminative models are ensembled into a final model (5)

a generative modeling approach is used to estimate the accuracies and correlations of the different labeling functions based on their observed agreements and disagreements; and finally, these accuracies are used to re-weight and combine the labels output by the labeling functions, producing a set of *probabilistic* (confidence-weighted) training labels.

In this work, we propose Osprey, a weak-supervision system, that builds on top of Snorkel framework [17] and extends it to support an end-to-end industrial ML deployment in three major ways (Figure 1): (i) We aim to democratize it to include non-programmer domain expert users. Instead of coding labeling functions, in Osprey domain experts inject business knowledge into the system through a new layer of higher-level interfaces. Moreover, with this new declarative layer we speed up the model development and tuning process (ii) By applying a synthetic rebalancing and augmentation technique, Osprey can handle a high class imbalance that is very common in practice. Such an imbalance makes hand-labeling training data prohibitively expensive *and* causes problems for existing weak supervision approaches (iii) Osprey uses a novel ensembling technique, wherein the generative model defined over the labeling functions is ensembled with the downstream discriminative models being weakly supervised, in order to support a generalization while keeping a high precision level. In Figure 1 we summarize the main additions to original Snorkel pipeline with orange highlighting.

We validate it on 3 real-world applications at Intel, where our ensembling technique yields improvements of over 10 precision points on an ablation, and the whole system achieves gains of 18.5 precision points and 28.5 coverage points over

```
RT @SMG CustomerPR: We are partnering with
@AnotherCompany to develop the #nextgeneration of
#AI. Ain't we lucky? :-) @SMG Customer
@AnotherCompanyCEO http://anothercompany.com/news
```

Figure 2: A tweet from a customer regarding a new partnership All underlined words are entities representing the same customer. Some of them are explicit Twitter handles, and others are anaphoric pronouns.

prior traditionally supervised and rules-based approaches. Furthermore, our approach is intended to be generic, and thus applicable to a range of other settings and domains.

In Section 2 we start by outlining a specific case-study involving relation-extraction over Twitter data, motivated by a deployment of Intel's Sales & Marketing Group (SMG). In order to motivate the weak supervision approach of Osprey, in Section 3 we provide a high-level analysis of the cost of these prior approaches, compared to weak supervision, using our experiences at Intel SMG. We then describe how we provide higher-level, more declarative weak supervision interfaces to non-programmer domain experts in 4. Next, in Section 5 we show how highly-imbalanced problems can be supported with intermediate datasets. We describe our approach to improving precision through a novel generative-discriminative model ensembling strategy in 6. We then present experimental details and results in Sections 7, 8 and conclude with a short review of related work in Section 9.

2 SALES & MARKETING - A CASE-STUDY

Intel's Sales & Marketing Group (SMG) is responsible for the company's interaction with its many customers. In order to optimize this interaction, SMG account managers need to be familiar with customers covered by them at all times. We study an application for monitoring Twitter in order to find publicly available business-related items about customers.

The high volume of tweets involving customers (millions per month) requires an automated process for their classification into one of several "business scenarios" defined by SMG domain experts. This modeling schema include classes such as "Partnership", "Merger & Acquisition" (*M&A*), "Product Launch", etc., but the number of business scenarios and their definitions evolve and change over time. This business-driven task faces the above mentioned challenges, which are common across many real-world problems and domains:

- *Extreme Class imbalance:* A preliminary analysis showed that the ratio of customer-related tweets relevant to any of the business-scenarios is 0.05% on average.
- *Prohibitive Labeling Cost:* With a positive ratio of 0.05%, directly developing a large-enough labeled training

set for any business scenario would be expensive, take many weeks/months, and require a full relabeling given any change to the schema.

The characteristics of Twitter as a medium and the business problem in general raise some additional points which are again instances of common themes present in many real-world settings involving complex, high-dimensional data:

- *Semantic Diversity:* Tweets come in many genres. Their content may consist of a formal/informal language, their syntax is often broken and they mix words, hashtags, emojis etc. Therefore, simple rules or pretrained models are rarely sufficient for a given business scenario, hence custom-trained ML models are required.
- *Data Drift:* Social-media language evolves rapidly over time. Hence labeled training sets require regular maintenance or complete replacement.
- *Precision-Oriented Workflow:* Account-managers prefer seeing fewer false positives, and receiving only the high-confidence relevant data items, especially since important events tend to resurface frequently.

The task of mapping customers to business-scenarios participation can be solved by determining for each customer-mention in a tweet, whether it participates one or more business-scenarios. Many times, in Twitter, explicit details are missing, for example, a tweet may describe a partnership between a customer and an unspecified company. Therefore, every business-scenario forms an independent *unary relation-extraction* problem rather than a binary one. In this paper we describe three approaches that represent standard approaches taken to the aforementioned problem, and to many other similar real-world ones: a rule-based approach, a weakly-supervised one, and a fully-supervised one, where we hold the choice of particular ML model constant between the latter two. We show that a very long and expensive tuning process is required for both the rule-based and the fully-supervised methods, and that the rule-based inference cannot be scaled easily. We then present `Osprey`, our weak supervision system for supporting non-coder domain experts, using intermediate synthetic datasets to assist in handling highly-imbalanced data, and using a novel weak supervision ensembling approach to improve precision.

3 REDUCING THE COST

In order to motivate the non-coder weak supervision approach taken by `Osprey`, we start by analyzing the costs for each of the three methods mentioned above, in terms of money and time, using our real-world industrial case study at Intel. We start by reviewing the high-level components of the different approaches, and in particular where they overlap and diverge. When analyzing the time cost, we focus on the human-driven component (e.g. expert's time spent on tuning rules) rather than on the machine-driven component (e.g. wall time for running training/inference). In our setting we found that the latter was negligible compared with the former. Moreover, it was was approximately constant across the weakly/fully supervised methods where model class and training procedure were fixed. The relation-extraction process is comprised of two steps - (i) recognizing customer entities; and (ii) verifying for each one whether it participates in a given relation type. For comparability, we use the same entity-recognition logic in all three methods. The entities of a tweet are all the Twitter handles (e.g. "@XYZ") and anaphoric pronouns within it, pointing to a customer (Figure 2). We leave other entity types for future work.

Next we describe each of the there methods at a high level for the purposes of the costs analysis. Additional details regarding each method will be given in the next sections.

3.1 Costs of a Legacy Rule-Based System

As the *unary relations* representing SMG's business-scenarios are not supported by public labeled datasets, SMG originally decided to develop a rule-based method. Directly compiling a labeled dataset for each business-scenario was ruled out as too expensive given the positive-class ratio of 0.05%.

The relational part of the model consists of several rule-groups for each business-scenario (Figure 3). A single rule is comprised of a basic pattern a.k.a "anchor" that is matched against each tweet. Once a basic match was found, it gets the *anchor*'s predefined score. Additional supporting / opposing patterns may be configured to raise / lower the match score. Following Figure 3 example, tweets containing "partnering" get a score of 0.8. If a "*supporter*" such as "excited" appears up to 4 words before the *anchor*, the score will be raised to 1.0. If the tweet also contains an "*opposer*" such as "years ago", the score will be reduced to 0.5. A relation will be emitted if the rule's final score$\geq$0.8 and the sum of group's final scores$\geq$1.0.

Overall, developing such a rule-based model involves three logical tasks: (i) finding enough positive and negative patterns; (ii) fine-tuning each pattern's weights, scores and thresholds; and (iii) adjusting the cross-pattern dependencies e.g. some positive patterns may provide weak signals separately but together when located close enough in text could be indicative enough.

In order to analyze the performance of this rule-based model, on a monthly-basis, all the relations predicted over the last month's tweets were sent to three Amazon Mechanical Turk (AMT) workers for validation (Figure 4). SMG domain experts then reviewed the relations that were unanimously approved or rejected by the AMT workers in order to curate and refine the model for the next month. During this iterative process, that took place for six month, SMG had to develop

```
"group-threshold" : 1.0
"scenario": "Partnership"
"rules" : [
  {
    "anchor": {
      "pattern": "partner(s|ing)?"
      "score" : 0.8
    },
    "threshold" : 0.9
    "supporters": [
      {
        "pattern": "excited|delighted"
        "max_distance": 4
        "direction": "before"
        "factor": 1.25
      }
    ]
    "opposers": [
      {
        "pattern": "(months|years) ago"
        "max_distance": -1
        "direction": "both"
        "factor": 2.0
      } ...
    ]
  }, ...
]
```

Figure 3: Legacy rules for the "Partnership" business scenario. All these thresholds and weights needed to be manually tuned and maintained.

an unsupervised mechanism for identifying rogue or low-quality AMT workers, whose outputs had a huge impact on the final accuracy. The model which detected abnormal workers was based on features such as the worker's label distribution, number of labels in the minority, speed, etc.

After six months of improving both the rule-based and the AMT management models, SMG's domain experts examined the latest results and found that the rule-based model provided relations with an average precision of 0.5. When combined with the AMT validation step, the average precision level was 0.85, so this was adopted as the full pipeline. With this pipeline, however, besides the money spent on a regular basis on AMT work, the process had to be applied in batches of at least 10 days in order to accumulate enough data statistics for the AMT worker validation system to function - a latency that lowered the business value of the resulting outputs. Furthermore, given shifting data and business targets that in our use-case require a retraining and hence a tuning process every few months, and with the brittle nature of this approach, this system proved difficult to maintain.

3.2 Costs of a Fully-Supervised System

Using a supervised ML model to achieve higher precision and recall on a task like relation extraction is a common and effective solution in practice today. The key ingredient in a standard, "fully-supervised" approach is a labeled training set, which for modern representation learning models must

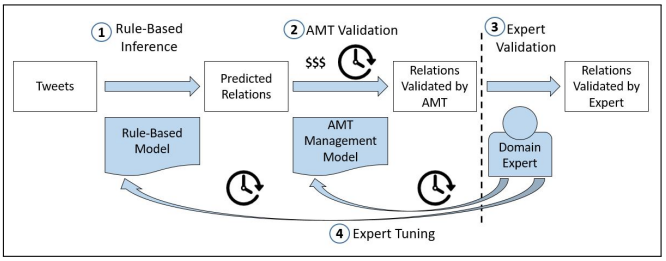

Figure 4: The iterative tuning process of the rule-based model and its costs. Rules-model and the AMT-management one are managed by the domain expert.

generally be quite large. In a highly class-imbalanced use case like ours, labeling and re-labeling such a dataset from scratch would have a huge cost. Therefore, instead, we effectively relied on the above mentioned rule-based pipeline to provide a high-recall preprocessing filter s.t. only data-items passing this filter were hand-labeled by AMT/experts (more details in Section 7). Hence, the traditional supervised approach in our class-imbalanced setting has a similar profile to that of the rule-based approach, given that the cost of training a model is negligible compared to that of creating the labeled training set through this process. In this Section, we continue our analysis of the high-level costs. We provide further details about the setup of the fully-supervised system in Section 7.

3.3 Costs of a Weakly-Supervised System

As mentioned in Sections 3.1, 3.2, developing a high-recall rule-based model is a crucial step for both the rule-based system (pre-AMT inference) and the fully-supervised one (pre-filter for manual labeling). In `Osprey`, however, the model's recall does not depend directly on the number of configured patterns but instead on the generalization power of Snorkel's discriminative model (step 4 in Figure 1), hence the time spent in `Osprey` on finding patterns (task i. of rules model development, Section 3.1) is much shorter. Moreover, by using Snorkel's unsupervised generative model to automatically estimate the accuracies of the labeling functions (*LFs*) [1, 17, 18], we find that we can skip the highly expensive fine-tuning done by expert (task ii.). Also, in `Osprey` instead of tuning the cross-pattern dependencies (task iii.), the system dynamically generates combinations of LFs (Section 4.1), while relying again on the generative model to automatically find their weights and accuracies. Finally, as the generative model of Snorkel provides highly-accurate labels (Table 5), there is no need to use AMT for validation, nor there is a need to develop or maintain an unsupervised AMT management model.

Another significant advantage of a weakly-supervised approach is that upon a data change necessitating a model

Table 1: Costs of different high level stages, in terms of human-time and money for rule-based (*"RB"*), weak-supervision (*"WS"*) and full-supervision (*"FS"*). Compute time is not included, as was negligible in comparison. Bold items indicate an extremely large cost.

	Labeling & Tuning		Ongoing Validation		Train & Inference
	Time	$	Time	$	Time
RB	**AMT, Expert**	**AMT**	**Batch latency**	**AMT**	
WS	Expert				
FS	**AMT, Expert**	**AMT**			

refresh (e.g. a language shift), a simple model re-training can be executed over a new *unlabeled* training set, with only few LFs generally needing to be amended before the retraining.

In the basic setup of Snorkel [17], LFs are usually described as being developed from scratch, on demand, by a domain expert. This generally mirrors other prior programmatic weak supervision pipelines. In this work however, we decouple the code layer from the configuration layer in order to support non-coder domain experts and democratize ML as well as to further reduce the experts workload in order to speed up end-to-end system development. Moreover, with Osprey's code-configuration decoupling, if a legacy rule-based model exists, it could be transformed automatically into a set of much "relaxed" LFs that are based solely on the patterns without any thresholds, weights, etc. (see Section 4.2) and the patterns development step (task i. Section 3.1) could be avoided. In real deployment, we find that such an automatic transformation is a good practice that can save significant time, and may additionally support useful backwards compatibility, e.g. Osprey can ingest the legacy rule-based model.

Table 1 summarizes the different costs within each system.

3.4 Costs Validation - a Human Study

In order to validate the above qualitative analysis, we have conducted a human study on four relation-extraction tasks. The domain experts in this study were three Intel product analysts that share the same business-group with the data scientists who developed Osprey. The study shows that on average, an Osprey model that outperforms both the legacy rule-based system and an equivalent supervised system can be developed by a domain expert in 1-2 weeks, where the iterative tuning step of these alternatives alone takes few months as described above. Moreover, a change to business definition requires an additional tuning of 2-3 weeks for the rule-based and fully-supervised approaches. In conclusion, from costs perspectives, a weak-supervision system offers significant practical advantages.

4 IMPROVED NON-CODERS SUPPORT

4.1 Decoupling Code and Configuration

In Snorkel, instead of manually labeling a large training set, domain experts compose relatively few code-snippets a.k.a *labeling functions* (*LFs*) capable of noisily labeling an unlabeled training set. Generally, given a data item, an LF returns a positive answer, a negative one or it abstains, but in this work all LFs are either positive ones (positive/abstain), or negative ones (negative/abstain).

Domain experts have a deep understanding of the business needs driving the ML application of interest. However, although being capable of creating simple LFs, they often struggle with composing complex LFs that require better programming skills. In this section we describe a higher-level interface provided by Osprey for non-programmers to specify weak supervision in our setting. This interface tackles this important practical gap by decoupling the domain understanding from the required coding skills. We also provide in this section more details on how this interface speeds up the end-to-end system development as mentioned above (Section 3.3) by avoiding AMT-validation, weights adjustment (task ii. in Section 3.1), and dependencies tuning (task iii. in Section 3.1).

Suppose a domain expert knows that "partners" is a positive term and "(years|months) ago" a negative regular expression pattern. In Osprey, rather than encoding this programmatically as in Snorkel, domain experts just enter these keywords or regular expressions in, along with the "polarity" information (pos/neg), into an Excel-spreadsheet based interface, and then Osprey auto-generates LFs. In more complex cases where in the rule-based system the user would had to manually specify and tune numeric thresholds—for example to express that "memorandum of understanding" within k words (for some k) of "partners" is a 2 times stronger positive indication — Osprey compiles a "dynamic combination" of LFs: a positive LF for "partners", a second positive LF for "memorandum of understanding" and few variants of positive LFs looking for both terms within m words for different values of m. In Osprey, domain experts are not required to tune weights, thresholds and scores of LFs since the generative model in Snorkel is capable of filtering out the noise by learning the accuracies of LFs, and re-weighting them appropriately [1, 17, 18]. Cross-pattern dependencies tuning is also avoided in Osprey by applying Snorkel's generative model on the dynamically created LF combinations. We find that Osprey's light and code-less configuration greatly reduces the time and complexity to configure and deploy an end-to-end system, as compared to the rules-based, fully-supervised and even the Snorkel baselines.

The LFs compilation process is supported in Osprey by a generic code layer of "LF templates" that is first developed

Business Scenario	Type	Polarity	Pattern
Partnership	Anchor	1	partner(s|ing)?
Partnership	Opposer	-1	(months|years) ago
Partnership	Supporter	1	excited|delighted

Figure 5: Osprey's Excel-like representation of the original "Partnership" rule (Figure 3). Unlike the rule-based system, Osprey's configuration requires no heavy tuning of thresholds, etc. Osprey's LF Generator uses a pattern template, which can be configured initially by a developer, to compile multiple LFs and possibly their dynamic combinations. Each LF is either positive or negative according to its polarity value.

by a developer, guided at a high-level according to business needs. Each template is an almost ready-to-use LF with some placeholders to be filled with user inputs.

A newly added "LF Generator" component reads multiple template configurations provided by the domain expert, injects the configured user-inputs (patterns in this case) into the appropriate code-logic previously developed (templates), and generates the final LF code on-the-fly (Figure 6).

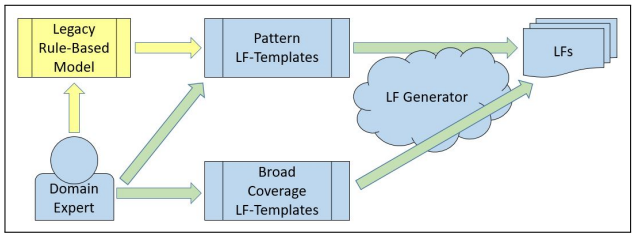

Figure 6: A domain expert configures the pattern-based and the broad-coverage LF templates through an Excel-like configuration. For backward compatibility with the legacy system, experts can configure the rule-based model and transform it automatically into Osprey's pattern-based configuration (yellow path, top-left). The LF Generator reads in all the configurations and compiles the final LFs by injecting the user inputs into the pre-coded templates logic.

While the resulting Excel spreadsheet-based interface is not "push-button", our main point is that it (i) enables non-programmers to quickly inject information, and (ii) decouples them from ML developers. In our experience at Intel, this has a fundamental impact on the way that ML systems are developed and deployed.

For backward-compatibility purposes, an existing rule-based model could be transformed into Osprey's simpler Excel-like configuration through an automated relaxation process where thresholds etc. are removed from the model.

4.2 Benefits of Decoupling

In the example above, the domain expert can add patterns to existing templates such as the pattern-template without any programming needed, and the LF Generator creates LFs for separate patterns and their combinations. With the code-configuration decoupling a developer can extend the LF Generator logic without having a deep domain knowledge, for example, to create from every entry in the pattern-LF configuration table two LF-variants – a first LF for a single appearance of this pattern, a second one for 2 or more appearances. This newly added logic, takes the choice of free parameters in the LFs—that otherwise, e.g. in the rule-based setting or basic Snorkel, a user would have to manually tune—and discretizes it, so that Snorkel's generative model can automatically handle the tuning. Note that directly tuning continuous parameters without discretization is an interesting direction for future work. However, this discretized approach seems to work well, and captures the intuition for example that the exact number of times that "partnership" appears does not matter much, but whether it appears once or many times might. Once this logic is added to the LF Generator code, the domain expert will get additional LFs "for free" without having to define new inputs for them. In this way, the domain experts, developers, and ML experts can all be cleanly decoupled within an organizational workflow.

Moreover, by decoupling interface layers in this way, domain-specific logic can easily be injected. For example, in our multiple relations problem, we can easily encode a rough prior that only one relation type will be present per tweet directly into the LF Generator - in other words, expressing a simple logical mutual exclusion constraint between LFs of different business-scenarios. Then, when building the LFs for a business-scenario R, the LF Generator not only forms positive-voting LFs from R's *anchors*, but also negative LFs from *anchors* of all business-scenarios $\overline{R}$ —again, all without additional input from the domain expert.

4.3 Broad Coverage LFs

In addition to pattern-based LFs, which are generally high-precision but low coverage, our system can also accept LFs that are high-coverage but lower precision, given the ability of Snorkel's generative model to re-weight these LFs accordingly [18]. Thus, the LF Generator in Osprey ingests another family of configurable templates that represent statistical features behaving differently in positive and negative samples. Many of these LFs, are also in line with the domain expert's rough intuition of how a general business tweet should look, and indeed they were added to Osprey upon

Type	Polarity	Value Range
Lowercase/Uppercase Ratio	-1	[[0.0, 0.5], [0.5, 1], [20, inf]]
Number of emojis in text	-1	[[1,2], [2,3], [3, inf]]
Sentiment	-1	[[-1.0, -0.5], [-0.5, -0.1]]

Figure 7: Osprey's configuration of some broad coverage LFs such as sentiment, ratio of lowercase/uppercase characters, and number of emojis. Matching LFs will be generated for all business-scenarios, as opposed to pattern-based LFs that are scenario-specific. Each template type (e.g. Sentiment) is backed by a pre-coded template logic loaded by the LF Generator.

the domain experts request. For example (Figure 7), a domain expert could express in LF that personal tweets will consist of more emojis than business tweets.

The exact statistical-characteristics behind these broad coverage LFs slightly change from one business-scenario to another. For example, "Partnership" tweets tend to be more formal and contain less emojis than "Conference Attendance" ones. However, by default, all scenarios share the same broad coverage LF templates and we rely on the the mostly-negative data and the generative model of Snorkel to handle these minor differences.

5 SYNTHETIC DATASETS

On preliminary experiments, we found out that even with the same LF generation technique (a core part of Osprey's contribution), "Vanilla" Snorkel fails to exceed an F1 score of 0.1 when trained over a dataset representing the natural distribution of data points coming directly from Twitter, due to its highly imbalanced nature (0.05%). An equivalent fully-supervised approach has failed to exceed this low score as well. In response, we propose a rebalancing approach that utilizes the logical structure of our multiple relation-classes problem to generate balanced synthetic datasets.

Many times in highly-imbalanced cases, a sub-sampling is used in order to place extra or less weight on different parts of the population. As our problem involves both highly-imbalanced data and multiple relation-classes, we take this approach one step further. For each business-scenario R, in order to differentiate better between items of R, items of other business-scenarios $\overline{R}$ and "general population", we construct three synthetic datasets: *Train*, *Dev*, and *Pre-Test* (Tables 2, 3). Each dataset is a mixture of the following logical groups: (i) General population candidates (ii) Approximately positive candidates from R; (iii) Approximately negative candidates from R; (iv) Approximately positives of $\overline{R}$ that form approximately negative candidates for R as the positive class ratio of every scenario is very low and business-scenarios do not tend to collide;. While "approximately positive" items for *Dev*, *Pre-test* are simply the relations validated by expert and found as positive (Figure 3), in the context of *Train*, approx. positives are relations filtered by the pattern-based LFs.

Table 2: Characteristics of each dataset. FS=full-supervision, WS=weak-supervision, RB=rule-based.

	Dev	Pre-Test	Train	Test
Distribution	synthetic	synthetic	synthetic	natural
Is Labeled?	partially	partially	no	no
WS-Usage	tuning	1st test	train	2nd test
FS-Usage	train	1st test	N/A	2nd test
RB-Usage	tuning	N/A	N/A	test

Table 3: Number of manually labeled items vs. number of candidates in each dataset

Business Scenario	Labeled	Dev	Pre-Test	Train	Test
Partnership	1176	23K	15K	80K	390K
Prod. Launch	1182	20K	13.5K	85K	390K
M&A	498	11K	7K	85K	390K

Surprisingly, even though a high class imbalance and a multi-class setting are each harder than a balanced binary case, sometimes, there are advantages in their combination especially if class-independence holds. We believe this approach can potentially generalize to other categorical settings especially in weak-supervision systems.

6 BETTER PRECISION WITH ENSEMBLES

In order to get high recall by generalizing to new data items while ensuring a high-level of precision, we examine several alternatives for the final weak supervision prediction model used in Osprey at test time: (i) Snorkel's discriminative model that is trained over the generative model's predictions. (ii) A bagging-like ensemble of discriminative models trained with different random seeds that control the items sampled for the training set. (iii) An ensemble in which the high-precision generative model—which is a model defined over the generally high-precision LFs—effectively provides a "safety-net" for the generalizing discriminative model, similar to how the high-precision AMT-workers provide a safety-net to rule-based model. We have tried various ensembling techniques for combining the generative and discriminative models, and eventually found that a simple approach that requires no heavy hyper-parameters tuning yielded the best results. In this approach, the ensembled prediction equals the discriminative prediction when the generative-prediction>0.5 or else the ensembled marginal is zeroed out.

Though ensembling is commonly used in many machine-learning systems, a generative-discriminative ensemble like

ours is very rare. Moreover, to our knowledge such an ensemble between a generative model and a discriminative model that was trained over it (thus already "captures its essense") is novel. In Section 7 we report gains of over 10 points in precision on an ablation for this generative-discriminative ensemble. We also report, that for a fixed precision-level, such an ensemble will generate gains of over 30 points in coverage over the discriminative model.

7 EXPERIMENTS SETUP

Examined Methods. We conducted a controlled experiment involving two systems: a weak-supervision one as described above (Figure 1), and an equivalent full-supervision version. The method we picked for the discriminative part of the two pipelines is a Bidirectional LSTM with an attention model, which is commonly used for text-related ML problems, and provides results close to the state of art (for example [22]).

The fully-supervised version was trained over *Dev*. Other less appealing alternative for a training set is the original labels validated in the rule-based tuning (right side of Figure 4), but this dataset is far too small (See Table 3) and our full-supervision system cannot train well over it (Figure 8). Another alternative is the much larger set of non-validated unanimous AMT answers (i.e. $\frac{0}{3}$ and $\frac{3}{3}$) but is much noisier. For comparability, the pattern-LFs of `Osprey` are based on the legacy rule-based patterns without any tuning. We also report the performance of the rule-based + AMT method, that is not scalable nor feasible, but still provides some notion of a human-driven base-line.

Raw Data and Candidates. For this work we used public tweets written in English, from 2017-2018, involving Intel customers. While these datasets cannot be shared due to business limitations, we plan to share a synthetic dataset as a follow-up work. As explained in Section 3, the relation-extraction (*RE*) process involves a preceding step of entities recognition shared between all methods. Hence, the reported results reflect only differences related to the "pure" *RE* logic.

Measurements. The highly-imbalanced data (0.05%) prevents the creation of a traditional labeled test set. For example, a test set of 500 positive items may require hand-labeling 1M items. Instead, for any method, we manually validated all the relations predicted over *Test* with predicted-probability>0.5. Overall, out of the 390K items in *Test*, 5-10K relations were validated for any single business-scenario.
After this manual validation, a method's precision can be easily estimated for any threshold>0.5. since both the true-positives (TP) and the false-positives (FP) are known. However, without all labels of *Test*, we cannot directly measure the recall since number of false negatives is unknown. Instead, we compare methods according to:

- *Relative-recall of method x w.r.t baseline b* - $\frac{|TP_x \cap TP_b|}{|TP_b|}$.
- *Relative-coverage of method x w.r.t to method y* - $\frac{|TP_x|}{|TP_y|}$ (similar to the relative-recall definition of [16])

8 RESULTS

Full-Supervision vs. Weak-Supervision. Table 4 describes the best results found for weak-supervision and full-supervision over 3 business scenarios. We can see that the weak-supervision system outperforms the full-supervision one in "Partnership", they are comparable over "Product Launch" and the weak-supervision wins by a knockout in the smaller business-scenario of "Merger & Acquisition". Moreover, in 2/3 business scenarios, the weak-supervision system is equivalent to or supersedes the legacy rule-based system which is backed by human (AMT) validation. In Figure 8 we can see that on the slightly-less imbalanced business-scenario of "Partnership" for which *Dev* is larger, full-supervision's performance is improved as more labeled samples are being used, but that requires again either manual labeling or rules-tuning.

Intermediate Results. Table 5 shows that the precision provided by the generative model over *Dev* is high and there was no over-relaxation of the deeply tuned legacy rules when transformed into the simple threshold-free model of `Osprey`.

Table 6 reports the performance of previously mentioned (Section 6) alternatives for `Osprey`'s final-model on *Test*. When fixing the number of TPs of every alternative to the baseline's, the precision levels of all weak-supervision models are lower than the baseline's. The bagging version of the discriminative model is more precise than the single-seed version, and the generative final model reaches a slightly higher precision level. However, when fixing the target precision-level to the value provided by the legacy system, the bagging version provides a much better relative recall and coverage than all other weak-supervision alternatives. Overall the results of every discriminative model are better when combined with the generative one.

9 FURTHER RELATED WORK

While `Osprey` is generic and could be applied to many domains, in this work it was validated through a relation-extraction problem over a highly imbalanced textual medium. Twitter data is widely used in research papers, but unlike our work, they tend to focus on fully-supervised approaches, text-level classification, and relatively balanced classes. For instance, a classification model for cyber-hate and inappropriate language over Twitter was built by [4], where 2K tweets were manually labeled for training and testing. [8] presented an algorithm for separating hate-speech from standard conversation and non-hate but offensive. This model too relies on a manually labeled corpus of 25K tweets, that are

Table 4: Performance of the weakly-supervised approach (with gen. model ensembling + bagging), vs. the high-cost fully-supervised equivalent, and the even higher-cost thus unfeasible rule-based baseline ("RB + AMT"). Measurements taken over *Test* after fixing either the number of TPs or the precision-level to the baseline's value. The Relative-Recall values indicate correlation with baseline's TPs and not the "absolute" recall. (*) = closest point to baseline's fixed value from which results are taken since only relations with marginal>0.5 were manually validated on *Test*. Bold = best results found with a supervised method in each business-scenario.

Business Scenario	Method	True Positive Fixed			Precision Level Fixed			
		TP	Precision	Relative Recall	Precision	TP	Relative Coverage	Relative Recall
Partnership	RB + AMT (baseline)	415	0.838	1	0.838	415	1	1
	Weakly-Supervised	415	**0.814**	**0.687**	0.838	**394**	**0.949**	**0.677**
	Fully-Supervised	415	0.635	0.682	0.838	148	0.355	0.319
Product Launch	RB + AMT (baseline)	200	0.473	1	0.473	200	1	1
	Weakly-Supervised	200	0.557	**0.610**	0.473	336	1.680	**0.770**
	Fully-Supervised	200	**0.606**	0.567	0.473	**375**	**1.873**	0.748
Merger & Acquisition	RB + AMT (baseline)	140	0.933	1	0.933	140	1	1
	Weakly-Supervised	140	**0.749**	**0.672**	0.924*	**85**	**0.607**	**0.491**
	Fully-Supervised	103*	0.325	0.553	0.933	5	0.036	0.038

Table 5: Performance of the generative model over *Dev*

Business Scenario	Precision	Recall
Partnership	0.881	0.766
Product Launch	0.894	0.686
Merger & Acquisition	0.855	0.718

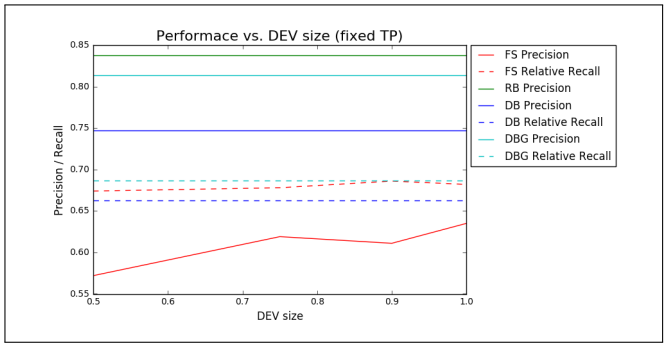

Figure 8: Performance of systems on "Partnership" when using subsets of *Dev* and fixing TPs number to legacy's one. FS=full-supervision, RB=rule-based, DB=weak-supervision's discriminative+bagging, DBG=DB+generative ensemble

somewhat imbalanced, with 5% hate-speech. [2] has taken a semi-supervised approach for relations-extraction using a bootstrapping method. The method was validated for 4 different relations over news documents. [10] extracts medicinal cause-effect relations from Twitter data, using syntactic dependencies between words. Twitter data is very often used for text-level sentiment-analysis (a special case of text classification) e.g. in [5] and [20] - both classify text-level sentiment rather than connect it to a specific target which is closer to *RE*. Weak supervision is also used for sentiment-analysis - [11] is using a deep-learning approach for binary sentiment classification of amazon reviews (well balanced datasets). [12] uses weak-supervision for *RE*, over news data with the novelty of capturing overlaps between relations. We cover general weak supervision related work in Section 1.

10 CONCLUSIONS

In the current work we have shown that highly class-imbalanced supervision problems can be addressed quickly, with low cost, and without domain experts needing programming skills, through a weakly-supervised system we propose, Osprey. In the setting we examine, rule-based systems and fully-supervised systems on the other hand are expensive, time-consuming and do not scale well. We have also provided a mechanism for an easy configuration of the weak-supervision model by decoupling the code-layer from the configuration-one and by doing so, we have not only added support for non-programmer domain-experts but reduced again the overall domain expert workload. We have seen that the performance of Osprey is much higher than the legacy rules-based and fully-supervised system in 2 out of 3 real-life business scenarios (and equivalent on the third one), both of which involved expensive and time-consuming expert validation. We believe that the new paradigm for non-programmer interaction with ML pipelines, encompassed by our system, can be applied to a range of rapidly-evolving, real-world applications both over twitter or text data and beyond.

Table 6: Ablation of different weakly-supervised (WS) pipeline variants as measured for "Partnership" on *Test* compared with the legacy rule-based baseline that involves AMT-validation. The results are reported after fixing either the number of TPs or the precision to the baseline's value. Bold = best results found by any weakly-supervised variant. Other business-scenarios behave similarly.

	True Positive Fixed			Precision Level Fixed			
Method	TP	Precision	Relative Recall	Precision	TP	Relative Coverage	Relative Recall
Rule-Based + AMT (baseline)	415	0.838	1	0.838	415	1	1
WS Gen.	415	0.795	0.673	0.838	24	0.058	0.054
WS Disc. Single-Seed (AVG)	415	0.592	0.647	0.838	72	0.172	0.152
WS Disc. Bagging	415	0.747	0.663	0.838	257	0.618	0.483
WS Disc. Single-Seed (AVG) + Gen.	415	0.699	0.672	0.838	355	0.855	0.632
WS Disc. Bagging + Gen.	415	**0.814**	**0.687**	0.838	**394**	**0.949**	**0.677**

REFERENCES

[1] Stephen H Bach, Bryan He, Alexander Ratner, and Christopher Ré. 2017. Learning the structure of generative models without labeled data. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 273–282.

[2] David S Batista, Bruno Martins, and Mário J Silva. 2015. Semi-supervised bootstrapping of relationship extractors with distributional semantics. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. 499–504.

[3] Jakramate Bootkrajang and Ata Kabán. 2012. Label-noise robust logistic regression and its applications. In *Joint European conference on machine learning and knowledge discovery in databases*. Springer, 143–158.

[4] Pete Burnap and Matthew L Williams. 2015. Cyber hate speech on twitter: An application of machine classification and statistical modeling for policy and decision making. *Policy & Internet* 7, 2 (2015).

[5] Prerna Chikersal, Soujanya Poria, and Erik Cambria. 2015. SeNTU: sentiment analysis of tweets by combining a rule-based classifier with supervised learning. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*. 647–651.

[6] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. 2018. AutoAugment: Learning Augmentation Policies from Data. *arXiv preprint arXiv:1805.09501* (2018).

[7] Nilesh Dalvi, Anirban Dasgupta, Ravi Kumar, and Vibhor Rastogi. 2013. Aggregating crowdsourced binary ratings. In *Proceedings of the 22nd international conference on World Wide Web*. ACM, 285–294.

[8] Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated hate speech detection and the problem of offensive language. *arXiv preprint arXiv:1703.04009* (2017).

[9] Alexander Philip Dawid and Allan M Skene. 1979. Maximum likelihood estimation of observer error-rates using the EM algorithm. *Applied statistics* (1979), 20–28.

[10] Son Doan, Elly W Yang, Sameer Tilak, and Manabu Torii. 2018. Using natural language processing to extract health-related causality from Twitter messages. In *2018 IEEE International Conference on Healthcare Informatics Workshop (ICHI-W)*. IEEE, 84–85.

[11] Ziyu Guan, Long Chen, Wei Zhao, Yi Zheng, Shulong Tan, and Deng Cai. 2016. Weakly-Supervised Deep Learning for Customer Review Sentiment Classification.. In *IJCAI*. 3719–3725.

[12] Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel S Weld. 2011. Knowledge-based weak supervision for information extraction of overlapping relations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*. Association for Computational Linguistics, 541–550.

[13] Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens van der Maaten. 2018. Exploring the Limits of Weakly Supervised Pretraining. *arXiv preprint arXiv:1805.00932* (2018).

[14] Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*. Association for Computational Linguistics, 1003–1011.

[15] Volodymyr Mnih and Geoffrey E Hinton. 2012. Learning to label aerial images from noisy data. In *Proceedings of the 29th International conference on machine learning (ICML-12)*. 567–574.

[16] Patrick Pantel and Marco Pennacchiotti. 2006. Espresso: Leveraging generic patterns for automatically harvesting semantic relations. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 113–120.

[17] Alexander Ratner, Stephen H Bach, Henry Ehrenberg, Jason Fries, Sen Wu, and Christopher Ré. 2017. Snorkel: Rapid training data creation with weak supervision. *Proceedings of the VLDB Endowment* 11, 3 (2017), 269–282.

[18] Alexander J Ratner, Christopher M De Sa, Sen Wu, Daniel Selsam, and Christopher Ré. 2016. Data programming: Creating large training sets, quickly. In *Advances in neural information processing systems*. 3567–3575.

[19] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. 2017. Revisiting Unreasonable Effectiveness of Data in Deep Learning Era. *CoRR* abs/1707.02968 (2017). arXiv:1707.02968 http://arxiv.org/abs/1707.02968

[20] Duyu Tang, Furu Wei, Nan Yang, Ming Zhou, Ting Liu, and Bing Qin. 2014. Learning sentiment-specific word embedding for twitter sentiment classification. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1. 1555–1565.

[21] Ce Zhang, Jaeho Shin, Christopher Ré, Michael Cafarella, and Feng Niu. 2016. Extracting databases from dark data with deepdive. In *Proceedings of the 2016 International Conference on Management of Data*. ACM, 847–859.

[22] Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. 2016. Attention-based bidirectional long short-term memory networks for relation classification. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, Vol. 2. 207–212.