# Witan

## Unsupervised Labelling Function Generation for Assisted Data Programming

Paper written by Benjamin Denham, Edmund M•—K. Lai, Roopak Sinha, M. Asif Naeem
Summarized and presented by Irina Klein, Remi Kalbe, Ryan Manthy

# 01 Introduction & Main concepts

- No initial supervision needed.
- Identifies patterns in unlabeled data.
- Users refine automated labeling rules.
- Approaches fully supervised method accuracy.
- Significantly reduces the manual labeling workload.

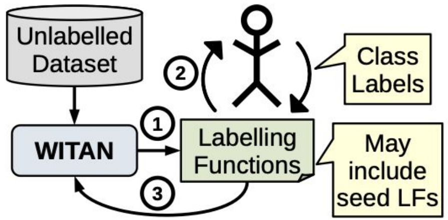
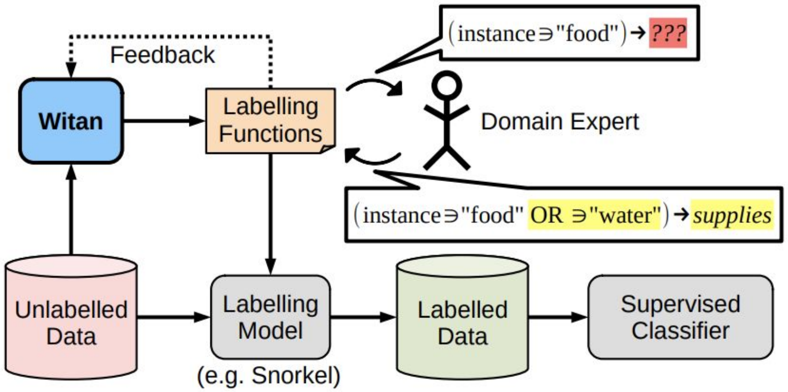



**Figure 1: Logical view of WITAN, demonstrating ① initial unsupervised generation of LFs, ② user review and assignment of class labels to LFs, and ③ extending an existing set of LFs.**

# 02 Context and motivation




- Machine learning advancements constrained by labeled data availability.
- Traditional LFs reduce manual labeling but are tedious and prone to bias.
- WITAN automates LF generation, enabling analysis of ambiguous data categories.
- Complements systems like Snorkel for quality training data refinement.

# 03 Data preparation

S1: “The movie was simply amazing with phenomenal effects, but a boring storyline.”

S2: The film's poor script was disappointing.

| Keywords | Sentence 1 | Sentence 2 | Sentence 3 | ... |
|---|---|---|---|---|
| Amazing | 1 | 0 | 0 | ... |
| Phenomenal | 1 | 0 | 0 | ... |
| Boring | 1 | 0 | 1 | ... |
| Poor | 0 | 1 | 0 | ... |
| ... | ... | ... | ... | ... |

# 04 The WITAN Algorithm

- Automated generation of labeling functions from binary features.
- Expert review to curate and assign appropriate labels to LFs.

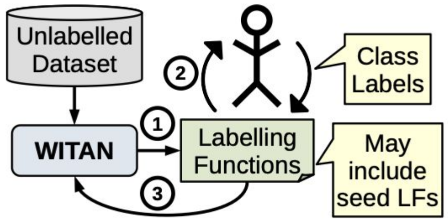
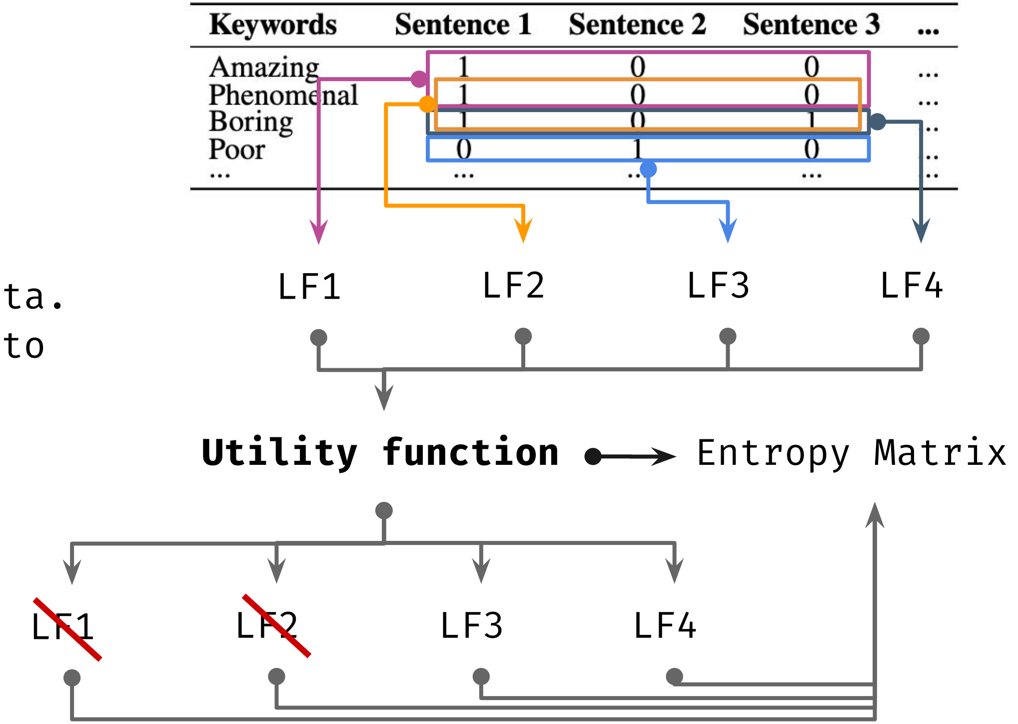



**Figure 1: Logical view of WITAN, demonstrating ① initial unsupervised generation of LFs, ② user review and assignment of class labels to LFs, and ③ extending an existing set of LFs.**

# 04 The WITAN Algorithm

Utility function: Introduction

- Smart selection based on pattern recognition.
- Reduces ambiguity in classifying unlabeled data.
- Identifies key patterns to unveil hidden data structures.



# 04 The WITAN Algorithm

Utility function: The Utility Function’s Calculations and Goals

1 2 3

$$IG\left(x_{*j}, c^{\lambda}\right) = H\left(x_{*j}\right) - H\left(x_{*j} \mid c^{\lambda}\right)$$

Information gain

Entropy of feature $x_{*j}$ before applying an LF

Conditional entropy of feature $x_{*j}$ given the application of a LF $\lambda$

| **Instance/Feature** | **Feature 1** | **Feature 2** | **Feature 3** | **Feature 4** |
|---|---|---|---|---|
| Instance 1 | $H_{1,1}$ | $H_{1,2}$ | $H_{1,3}$ | $H_{1,4}$ |
| Instance 2 | $H_{2,1}$ | $H_{2,2}$ | $H_{2,3}$ | $H_{2,4}$ |
| Instance 3 | $H_{3,1}$ | $H_{3,2}$ | $H_{3,3}$ | $H_{3,4}$ |
| Instance 4 | $H_{4,1}$ | $H_{4,2}$ | $H_{4,3}$ | $H_{4,4}$ |

Table 1: Example of an Entropy Matrix ($\bar{H}$) for the WITAN Algorithm. Each cell $H_{i,j}$ represents the entropy of Feature $j$ for Instance $i$.

# 04 The WITAN Algorithm

## Utility function: The Utility Function's Calculations and Goals

1 2 3

$$U_{X,\bar{H},\gamma,w}(\lambda) = \sum_{\substack{j=1,\\ j\notin d^{\lambda}}}^{m} w_j \sum_{\substack{i=1,\\ c_i^{\lambda}=1}}^{n} \max(0, \bar{h}_{i,j} - H(x_{*j}|c^{\lambda}))^{\gamma}$$

Utility function sums the information gain for the features not already covered by the LF by taking the dataset, entropy matrix, information gain, wright vector, and LF set as arguments.

$w$ is the weight vector measuring information gain for each feature. This creates a rank of features that influences label prediction. The sum across all features adjusts the utility function to emphasize relevance.

The max() function is written to compare initial entropy with the post-LF application entropy of the feature. This leverages the LF to reduce uncertainty. Then IG is scaled by $\gamma$ to highlight features with greater information reduction.

### Function Objectives

Labeling functions(LFs) are ranked based on their ability to improve data classification.

### Emphasis on Information Gain

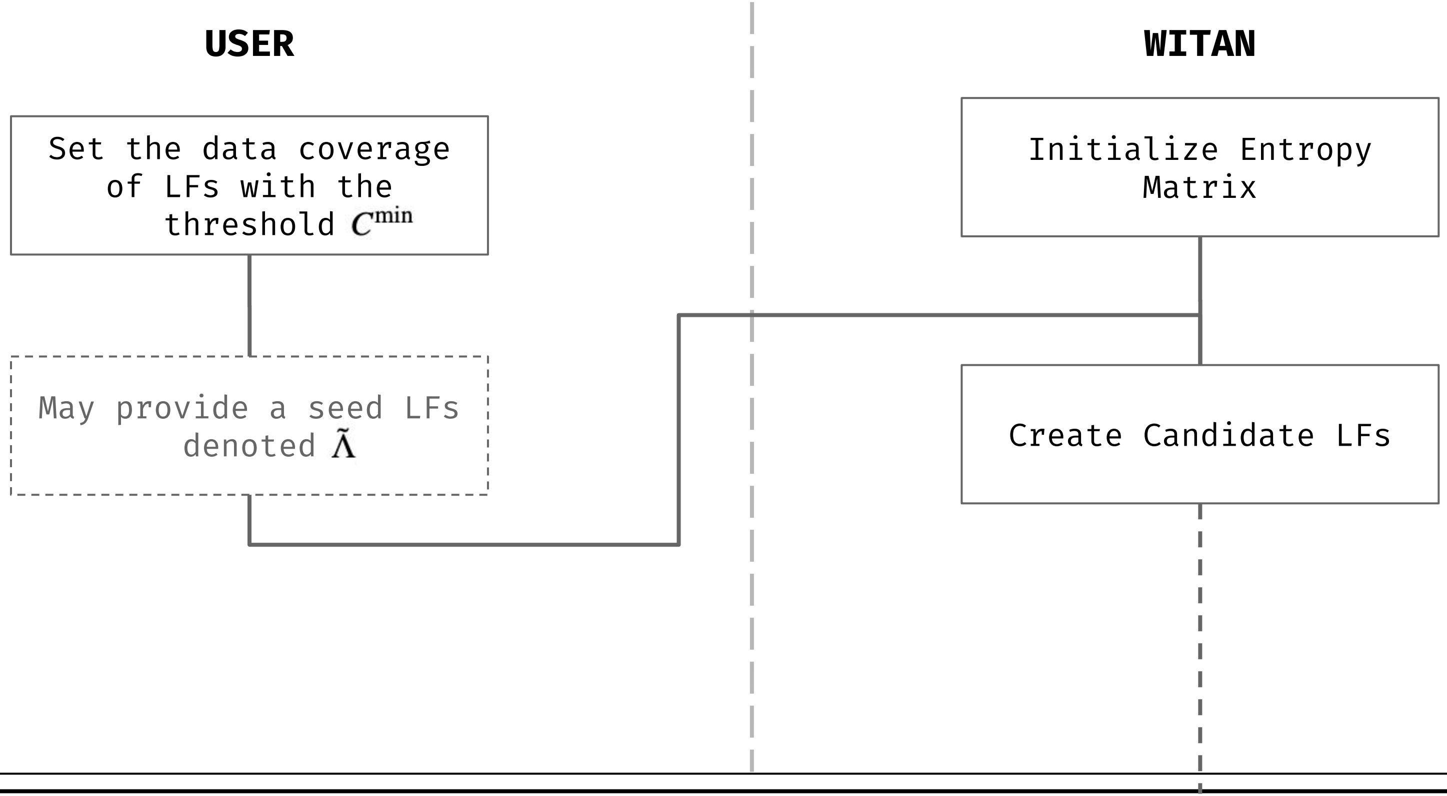
Information gain is prioritized by the Utility Function. This means the LF's ability to reduce uncertainty influences will be prioritized.

### Mechanisms

LFs that consistently improve predictions across the entire data set are prioritized by the Utility Function.

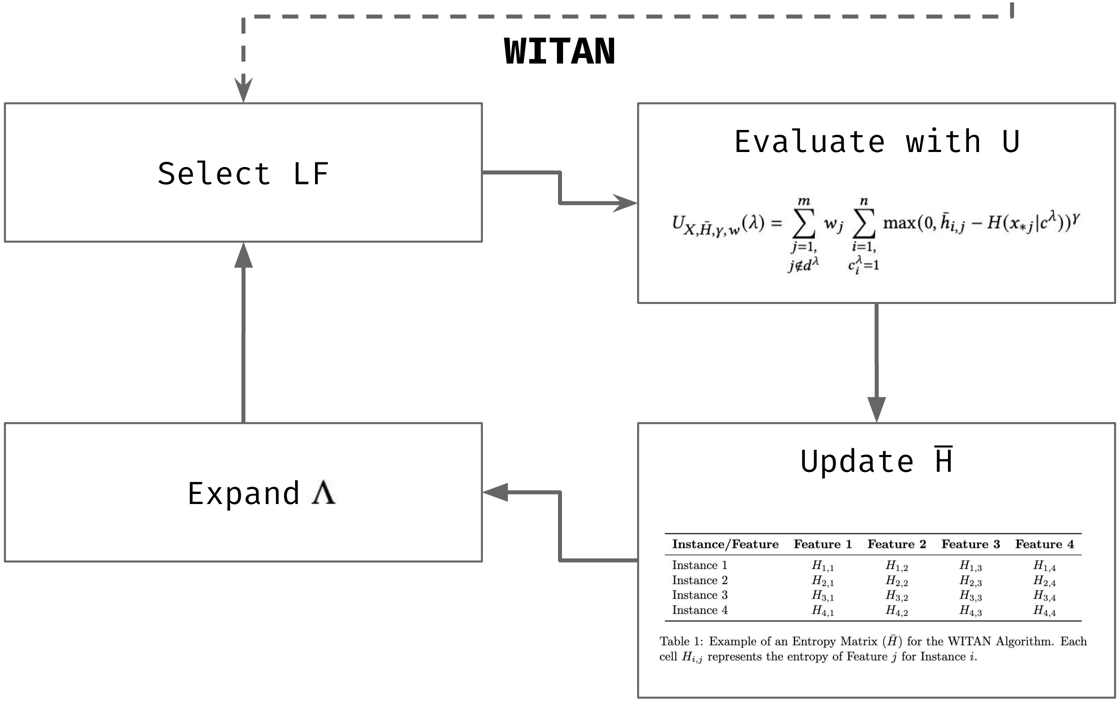
# 04 The WITAN Algorithm

Core of the algorithm



# 04 The WITAN Algorithm

Core of the algorithm




| Instance/Feature | Feature 1 | Feature 2 | Feature 3 | Feature 4 |
|---|---|---|---|---|
| Instance 1 | $H_{1,1}$ | $H_{1,2}$ | $H_{1,3}$ | $H_{1,4}$ |
| Instance 2 | $H_{2,1}$ | $H_{2,2}$ | $H_{2,3}$ | $H_{2,4}$ |
| Instance 3 | $H_{3,1}$ | $H_{3,2}$ | $H_{3,3}$ | $H_{3,4}$ |
| Instance 4 | $H_{4,1}$ | $H_{4,2}$ | $H_{4,3}$ | $H_{4,4}$ |

Table 1: Example of an Entropy Matrix ($\bar{H}$) for the WITAN Algorithm. Each cell $H_{i,j}$ represents the entropy of Feature $j$ for Instance $i$.

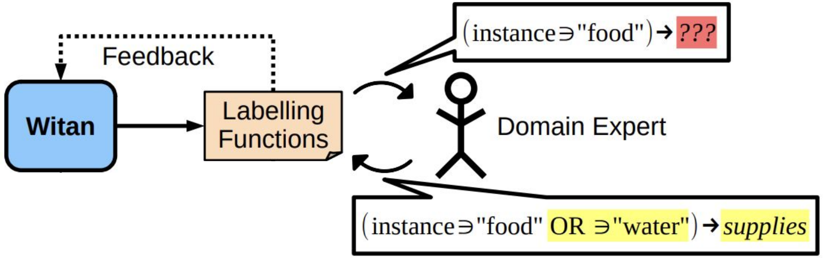
# 04 The WITAN Algorithm

Core of the algorithm

- User marks approved LFs
- Feedback updates weights of LFs (if provided)
- User decides class label meanings for selected LFs
- User may manually improve generated conditions

| | ∋ help | ∋ need | ∋ food | ∋ like | ∋ know | ∋ water | ∋ information | ... |
|---|---|---|---|---|---|---|---|---|
| **help** | 0 | 0 | 0 | 25 | 28 | 0 | 125 | ... |
| **need** | 0 | 0 | 0 | 191 | 11 | 0 | 0 | ... |
| **food** | 0 | 0 | 0 | 85 | 45 | 0 | 116 | ... |
| **like** | 27 | 184 | 84 | 0 | 0 | 78 | 0 | ... |
| **know** | 29 | 12 | 43 | 0 | 0 | 38 | 0 | ... |
| **water** | 0 | 0 | 0 | 72 | 36 | 0 | 52 | ... |
| **information** | 104 | 0 | 95 | 0 | 0 | 48 | 0 | ... |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋱ |
| $U =$ | 253 | 373 | 611 | 594 | 248 | 408 | 693 | ... |

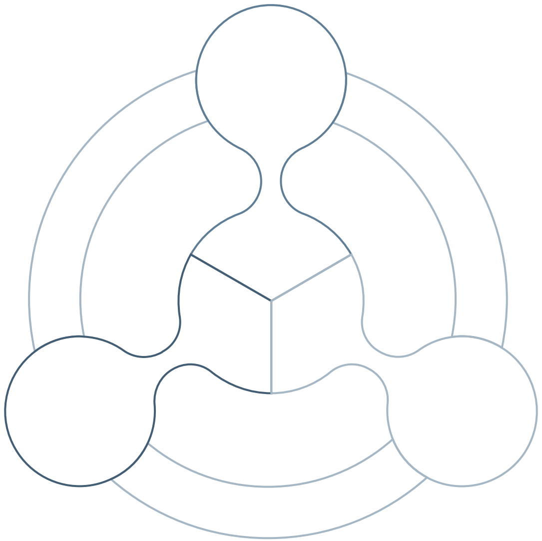
The LF condition with **highest utility** is selected next to propose to the user. E.g. They may assign a class *advice* to complete the LF: ∋ information → *advice*



# 04 The WITAN Algorithm

Flexibility and Adaptation in WITAN's Algorithm

- Feature weights recalibrated post-LF selection for enhanced accuracy.
- Candidate LF pool dynamically updated after each round.
- System extensibility for integrating new insights and dataset variations.



# 05 WITAN in action

## Binary Classification Performance Review

**Table 3: Binary classification F1 scores for unseeded and seeded labelling methods at 25 and 100 interactions (IC).**

| Method | IC | IMD | IMG | BPA | BPT | BJP | BPP | AZN | YLP | PLT | FNS | BDB | BAG | ATW | DMG | SPM |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Full supervision | | 0.836 | 0.780 | 0.950 | 0.898 | 0.933 | 0.942 | 0.905 | 0.872 | 0.779 | 0.976 | 0.995 | 0.901 | 0.822 | 0.964 | 0.941 |
| WITAN | 25 | 0.727 | **0.710** | **0.923** | **0.848** | **0.892** | **0.860** | **0.772** | **0.737** | **0.675** | **0.900** | **0.979** | **0.777** | 0.510 | 0.723 | **0.816** |
| | 100 | 0.753 | 0.768 | 0.931 | 0.817 | 0.887 | 0.875 | 0.779 | 0.795 | 0.624 | 0.905 | 0.949 | 0.708 | 0.550 | 0.690 | 0.781 |
| WITAN-Core | 25 | **0.738** | 0.703 | 0.910 | 0.836 | 0.859 | 0.840 | 0.760 | 0.737 | 0.667 | 0.891 | 0.978 | 0.728 | **0.632** | 0.783 | 0.745 |
| | 100 | **0.785** | **0.771** | **0.938** | 0.833 | **0.900** | 0.848 | **0.834** | **0.801** | 0.719 | **0.908** | **0.976** | 0.764 | 0.594 | 0.846 | 0.843 |
| IWS-AS | 25 | 0.363 | 0.511 | 0.619 | 0.398 | 0.504 | 0.660 | 0.559 | 0.427 | 0.635 | 0.448 | 0.774 | 0.479 | 0.536 | 0.676 | 0.536 |
| | 100 | 0.575 | 0.746 | 0.815 | 0.477 | 0.845 | 0.835 | 0.796 | 0.565 | **0.721** | 0.391 | 0.965 | 0.688 | 0.607 | 0.808 | 0.805 |
| IWS-LSE-AC | 25 | 0.000 | 0.354 | 0.793 | 0.524 | 0.640 | 0.839 | 0.574 | 0.398 | 0.467 | 0.375 | 0.938 | 0.525 | 0.526 | 0.634 | 0.675 |
| | 100 | 0.656 | 0.594 | 0.850 | 0.509 | 0.826 | 0.850 | 0.790 | 0.651 | **0.721** | 0.476 | 0.964 | 0.668 | 0.604 | 0.809 | 0.805 |
| Snuba | 25 | 0.458 | 0.436 | 0.718 | 0.726 | 0.693 | 0.820 | 0.507 | 0.497 | 0.601 | 0.584 | 0.781 | 0.489 | 0.519 | 0.781 | 0.624 |
| | 100 | 0.501 | 0.521 | 0.844 | 0.763 | 0.759 | 0.830 | 0.651 | 0.668 | 0.631 | 0.796 | 0.862 | 0.647 | 0.503 | 0.783 | 0.598 |
| HDC | 25 | 0.000 | 0.388 | 0.663 | 0.746 | 0.688 | 0.802 | 0.576 | 0.495 | 0.596 | 0.788 | 0.753 | 0.749 | 0.403 | **0.865** | 0.777 |
| | 100 | 0.545 | 0.428 | 0.686 | 0.746 | 0.695 | 0.862 | 0.572 | 0.500 | 0.714 | 0.836 | 0.759 | **0.806** | 0.442 | 0.894 | 0.452 |
| Semi-supervised | 25 | 0.543 | 0.464 | 0.664 | 0.609 | 0.743 | 0.450 | 0.521 | 0.468 | 0.536 | 0.796 | 0.861 | 0.436 | 0.450 | 0.502 | 0.575 |
| | 100 | 0.607 | 0.599 | 0.551 | 0.567 | 0.491 | 0.590 | 0.700 | 0.467 | 0.558 | 0.642 | 0.913 | 0.592 | 0.546 | 0.789 | 0.571 |
| Active learning | 25 | 0.551 | 0.562 | 0.846 | 0.739 | 0.739 | 0.860 | 0.622 | 0.596 | 0.583 | 0.799 | 0.894 | 0.519 | 0.568 | 0.761 | 0.728 |
| | 100 | 0.666 | 0.642 | 0.917 | **0.837** | 0.845 | **0.911** | 0.787 | 0.745 | 0.684 | 0.895 | 0.973 | 0.780 | **0.715** | **0.914** | **0.876** |
| Seeded WITAN | 25 | **0.774** | **0.761** | **0.931** | 0.834 | **0.890** | **0.880** | 0.742 | **0.795** | **0.696** | **0.909** | **0.980** | **0.768** | 0.544 | 0.759 | 0.696 |
| | 100 | 0.779 | 0.748 | 0.931 | 0.806 | 0.880 | **0.881** | 0.808 | **0.802** | 0.616 | 0.903 | 0.960 | 0.726 | 0.570 | 0.573 | 0.696 |
| Seeded WITAN-Core | 25 | 0.756 | 0.756 | 0.926 | **0.840** | 0.863 | 0.838 | **0.785** | 0.768 | 0.680 | 0.893 | 0.974 | 0.756 | **0.654** | **0.792** | **0.767** |
| | 100 | **0.790** | **0.763** | **0.941** | **0.822** | **0.900** | 0.861 | **0.835** | 0.796 | 0.718 | **0.915** | **0.976** | **0.770** | 0.595 | **0.846** | **0.843** |
| Seeded IWS-AS | 25 | 0.703 | 0.657 | 0.752 | 0.576 | 0.644 | 0.763 | 0.646 | 0.594 | 0.678 | 0.649 | 0.899 | 0.600 | 0.645 | 0.716 | 0.637 |
| | 100 | 0.771 | 0.721 | 0.912 | 0.606 | 0.849 | 0.836 | 0.810 | 0.736 | **0.721** | 0.522 | 0.968 | 0.687 | 0.607 | 0.805 | 0.806 |
| Seeded IWS-LSE-AC | 25 | 0.333 | 0.393 | 0.877 | 0.652 | 0.705 | 0.842 | 0.632 | 0.443 | 0.659 | 0.521 | 0.942 | 0.610 | 0.621 | 0.761 | 0.725 |
| | 100 | 0.613 | 0.457 | 0.904 | 0.602 | 0.834 | 0.849 | 0.784 | 0.721 | **0.721** | 0.750 | 0.967 | 0.686 | **0.607** | 0.821 | 0.806 |
| Seeded CBI | 25 | 0.554 | 0.596 | 0.708 | 0.672 | 0.619 | 0.715 | 0.510 | 0.594 | 0.525 | 0.740 | 0.722 | 0.593 | 0.428 | 0.497 | 0.620 |
| | 100 | 0.617 | 0.603 | 0.796 | 0.744 | 0.781 | 0.778 | 0.507 | 0.653 | 0.525 | 0.809 | 0.926 | 0.620 | 0.428 | 0.497 | 0.620 |

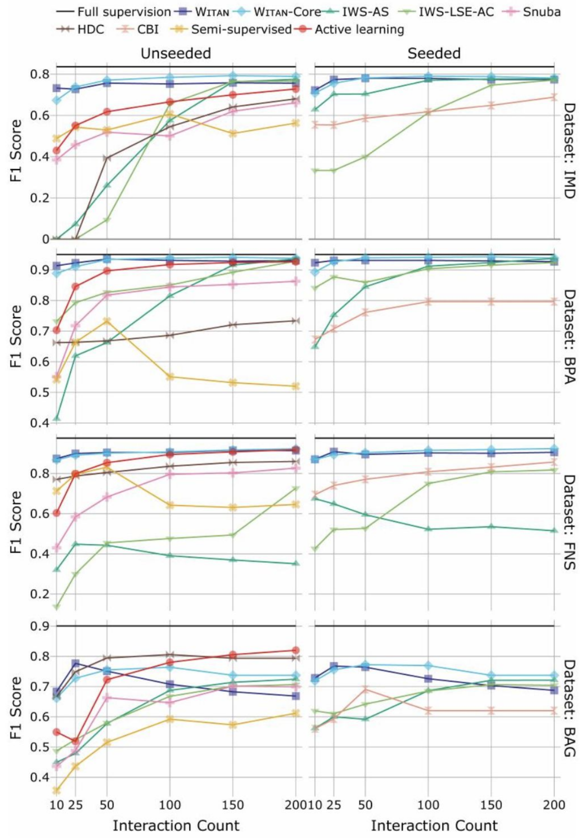



**Figure 4: Binary classification results.**

# 05 WITAN in action

Multi-Class Classification Performance Review

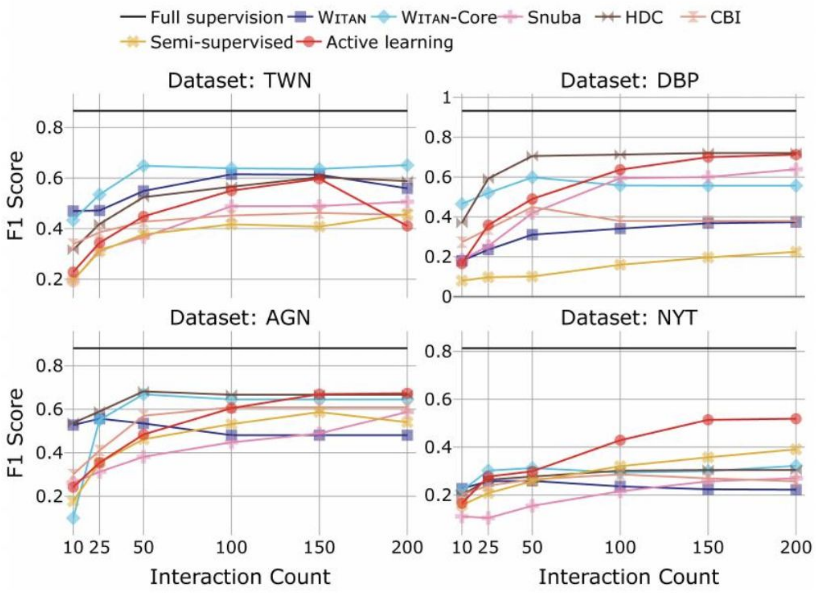
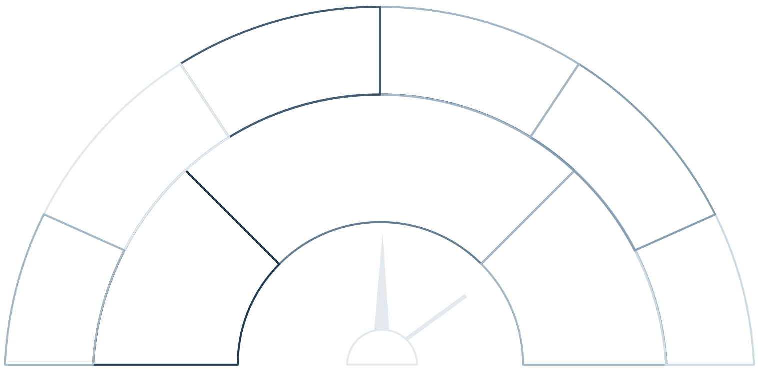



**Figure 5: Multi-class classification results.**

# 05 WITAN in action

WITAN's Performance and Critical Evaluation

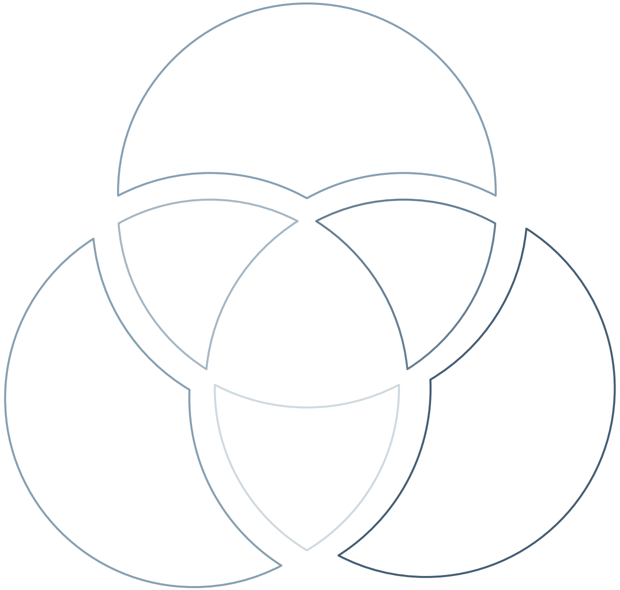
- Interactive Efficiency: Quick results with minimal user time.
- Versatility: Adapts to a wide array of datasets and tasks.
- Performance Trends: Achieves near-optimal outcomes expediently.
- Constraints: Challenges with late-stage LFs and class balance.
- Quality-Quantity Balance: The equilibrium necessary for success.



# 05 WITAN in action

## Ethical Considerations & Bias

- Mitigating Biases: Critical in maintaining label accuracy and trust.
- Ethical Labor Practices: A shift towards automation for ethical improvement.
- Dual Considerations: Balancing challenges with potentials in automation.



# 05 WITAN in action

## Running WITAN

**IMDb Reviews**

In [22]:
```
dataset = 'imdb'
witan_rule_browser(dataset_results[dataset], browser_args[dataset])
```

**negative:** waste / worst (coverage: 13%, accuracy: 90%)
documentary
movie
    **negative:** waste / bad / stupid / crap (coverage: 22%, accuracy: 79%)
    **negative:** horrible (coverage: 3%, accuracy: 88%)
**positive:** wonderful / excellent / superb (coverage: 14%, accuracy: 80%)
films
animation
episodes / episode
**positive:** loved (coverage: 5%, accuracy: 75%)
rent
recommend
funny
war
**negative:** crap / awful / lame (coverage: 11%, accuracy: 87%)
enjoyed / great / relationship / young / highly / beautiful
jokes
horror
supporting
**negative:** avoid (coverage: 3%, accuracy: 82%)

- Available at https://github.com/ben-denham/witan
- Well documented and easy to run using Docker.
- Simulated user which selects LFs having accuracy at least 20% above the random chance.

# 05 WITAN in action

## Running WITAN

### 20Newsgroups Topics

In [30]:
```
dataset = 'twentynews'
witan_rule_browser(dataset_results[dataset], browser_args[dataset])
```

nntp
- **computer:** thanks (coverage: 15%, accuracy: 57%)
  - **computer:** advance (coverage: 3%, accuracy: 68%)

article
  - **religion:** rutgers (coverage: 3%, accuracy: 87%)
  - **sports:** dod (coverage: 2%, accuracy: 92%)

- **sports:** bike (coverage: 2%, accuracy: 100%)
- **computer:** hi / windows / pc / graphics / dos / mac (coverage: 18%, accuracy: 80%)
- **sports:** hockey / season / baseball (coverage: 6%, accuracy: 94%)
- **science:** clipper (coverage: 3%, accuracy: 97%)
- **religion:** god / christians / christian / jesus / bible / christianity / religion / waco / jews / christ / religious / church / israel (coverage: 18%, accuracy: 56%)

cwru / uiuc / cmu
- **computer:** hello (coverage: 2%, accuracy: 59%)
- **sports:** games (coverage: 3%, accuracy: 84%)

netcom
- **sports:** car (coverage: 5%, accuracy: 74%)

wondering
- **computer:** appreciated (coverage: 4%, accuracy: 58%)

1993apr20 / writes
  - **sports:** team (coverage: 3%, accuracy: 91%)

# 06 Conclusion

- Feature weights recalibrated post-LF selection for enhanced accuracy.
- Candidate LF pool dynamically updated after each round.
- System extensibility for integrating new insights and dataset variations.

# Q&A