Author Verification: Basic Stacked Generalization Applied To Predictions from a Set of Heterogeneous Learners

Erwan Moreau, Arun Jayapal, Gerard Lynch and Carl Vogel

CNGL & Trinity College Dublin

moreaue@cs.tcd.ie, jayapala@cs.tcd.ie, gerard.lynch@ucd.ie, vogel@cs.tcd.ie

This research is supported by Science Foundation Ireland (Grant 12/CE/I2267) as part of the Centre for Next Generation Localisation (www.cngl.ie) funding at Trinity College, University of Dublin.

PAN 2015

Approach

- Regression problem (at the dataset level)
  - one instance = one problem (known docs + unknown doc)
  - optimize **AUC** $\times$ **c@1**
- Combining multiple learners
- Genetic algorithm used to:
  - train the individual learners,
  - train the meta-model.

- Experience from PAN'2014:
  - Genetic algorithm: tends to overfit
  - Two approaches:
    - *Fine-grained:* many parameters to maximize performance
    - *Robust:* basic approach to avoid overfitting

  → strategy chosen manually by dataset
- Results obtained by the organizers meta-model:

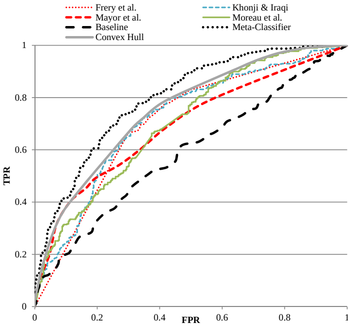


Fig. 1. ROC graphs of the best performing submissions and their convex hull, the baseline method, and the meta-classifier.

Strategies

1. Fine-grained strategy: many parameters, maximize performance
2. Robust strategy: basic approach, safer
3. General Impostor
   - Idea: meta-comparison against third-party documents
   - Used by best system at PAN'14
4. Topic modelling
   - Modified for *style* distinctiveness
   - Goal = Complementarity
5. Universum Inference
   - Bootstrapping method
   - Homogenity of documents snippets mixed together

Configurations

- Representing distinct set of parameters in an homogeneous way
- Set of key-value pairs: $C = \{p_1 \mapsto v_1, \ldots, p_n \mapsto v_n\}$
- Describe the meta-parameters of a strategy
  - In training mode, a configuration C and a set of instances (problems) S define a model M in a unique way:
    $$f_{train}(C, S) = M$$
  - In testing mode, a configuration C, a model M and an instance s define a unique prediction:
    $$f_{test}(C, M, s) = p$$
- Specific set of parameters for each strategy
- Very large space of possible configs

Common to all strategies

- Low-level features: various kinds of *n*-grams
  - words, letters, POS tags, skip-grams...
- Output of the strategy: a set of *indicators* (high-level features)
- Regression algorithm $\rightarrow$ score in $[0, 1]$
  - SVM regression, Decision trees regression
- Optional: classification to try to detect ambiguous cases
  - Uses indicators + predicted score
  - Optimize C@5 score

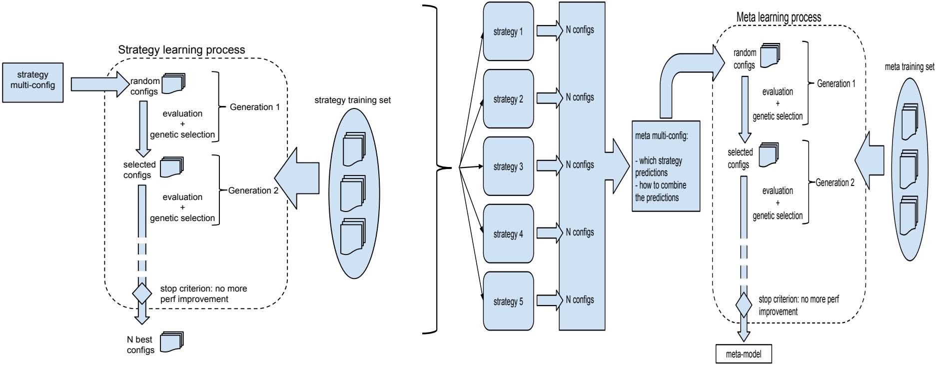
Genetic Algorithm

- A *multi-configuration* associates multiple values to one parameter:

$$MC = \left\{ p_1 \mapsto \{v_1^1, \ldots, v_{m_1}^1\}, \ldots, p_n \mapsto \{v_1^n, \ldots, v_{m_n}^n\} \right\}$$

- 1 configuration = 1 "individual"
- Multi-configuration = space of all combinations = input
- Basic genetic process:
  - first generation initialized randomly
  - Then selection based on previous generation performance
  - Possibility of mutation.
- Selects a subset of optimal configurations for each strategy

Architecture



ML Setting

Risk = overfitting

- Genetic process: *inner* k-fold CV
  - New k-partitioning at every generation
- Chained sequences with k increased
- Final 10×2 CV
  - Control the influence of k-partitioning

Hybrid setup

- Training set split into:
  - Strategy training: 50% instances
  - Meta-stage training: 25%
  - Meta test set: 25%

\+ Final eval with bagging

\+ Overall 2-fold CV

Results

Dataset	Meta test set	Full training set	Test set perf.	Test set rank
Dutch	0.710	0.722	0.635	1st
English	0.405	0.421	0.453	6th
Greek	0.656	0.761	0.693	2nd
Spanish	0.950	0.952	0.661	4th
		Macro-average	0.610	2nd

- Influence of the size of the sample
  - English: only one known doc by case
  - Spanish: four known docs by case
- Similar perf on training and test set
  - no overfitting *(except with Spanish)*

Conclusion and future work

- Combining heterogeous learners works well
- Works better with more information
- Selecting learners based on diversity?

- In progress: making the code available