A Slightly-modified GI Author-verifier with Lots of Features (ASGALF)

Mahmoud Khonji, Youssef Iraqi
{mkhonji, youssef.iraqi}@ku.ac.ae

Khalifa University, UAE

Outline

- General Impostors (quick intro; our imp.)
- Score aggregation.
- Features.
- Parameter tuning.
- Stuff that are possibly limitations of our classifier.

GI (quick intro reflecting our imp.)

score = 0
general_impostors(*knowns*, *unknown*):
    n = |*knowns*|
    forall *known* in *knowns*:
        score += **impostors**(*known*, *unknown*) / *n*

if *score* > *threshold:*
    return “same”
else
    return “notsame”

```
impostors(known, unknown):
    score2 = 0
    for 1 ... runs_num:
        imps = get_imps_rnd(lang-genre-docs, n)
        fs = get_fs_rnd(features, f)
        best_imp_to_known
        best_imp_to_unknown
        forall imp in imps:
            sim_k = sim(imp, known)
            sim_u = sim(imp, uknown)
            best_imp_to_known = imp if higher sim
            best_imp_to_unknown = imp if higher sim
```

```
        if   sim(known, unknown)^2 >
                sim(sim_k, known) * sim(sim_u, unknown):
                    score2 += 1/runs_num

return score2
```

Score aggregation

Instead of:

if **x** > **y**:

score2 += 1/*runs_num*

We did:

score2 += **x/y**

Features

All n-grams that have occurred at least 5 times in any document.

$n \in \{1, ..., 10\}$

gram $\in$ {letters, words, words_function, words_shape, words_post, words_post-word}

Features examples

words_functions:
If x, y, and z are function words in “x .... y .... z ...”, then a 2-gram would be {x:y, y:z}.

words_post:
“saw the saw” would become “VBD DT NN”, then a 2-gram set would be {VBD:DT, DT:NN}

words_post-word:
“saw the saw” would become “saw-VBD the-DT saw-NN”, then a 2-gram set would be {saw-VBD:the-DT, the-DT:saw-NN}

Parameter tuning

Assuming *threshold* = 0.5, apply a correction to the score to maximize accuracy.

First, find optimal *threshold (*exhaustively). One that maximizes accuracy on training set.

Then, *correction = 0.5 - threshold.*

Stuff that are possibly limitations

- Not fully taking advantage of C@1.
- Parameters are not found rigorously (a few manual trials).
- Using min-max might not show some interesting patterns.
- Being too-spoiled by **impostors** robustness against noisy features (using too many features slowed our implementation while possibly not adding much value)
- The usual things: clumsy code.

Acknowledgement

Thanks to Shachar Seidman for answering our questions about GI.

Thank you