# Fake Italian Restaurants evaluator (FIRe)

By: Andrea Piolini, Matteo Bucalossi, Srijon Mukhopadhyay

# GOAL:

**Create a program that verifies Italian restaurants' authenticity by identifying poorly written menus of Italian restaurants in the U.S. and assigning a score to them**

# Has this been done before?

- Cross-spelling check is a relatively new and unexplored branch of NLP.
  - Literature lacks work on non-native speakers' text analysis.
- Italian NLP community has focused on more "classic" NLP tasks (i.e. information retrieval) .
  - The University of Pisa is the main Italian NLP hub.
  - The field of Italian spelling and grammar check is more elusive and complex, and it's currently under researched.
- Menu retrieval and comparison is also a relatively new and unexplored branch of NLP.

# Steps:

1. Gather the data to create two corpora:
    a. Background Corpus: contains the correct Italian words scraped from online Italian food recipe websites and Italy-based Italian restaurants' menus
    b. Test Corpus: contains words scraped from menus of Italian restaurants based in the U.S.
2. Compare the two corpora to identify whether there are poorly written Italian words in the test corpus.
3. Assign an "authenticity score": 1 to 5 (5 being the most authentic) depending on the number of mistakes.

# Authenticity Score

| Number of Mistakes | Score | Label |
|---|---|---|
| 0-1 | 5 | IGA ( Italian Grandma-Approved) |
| 2-3 | 4 | Second Generation |
| 4-5 | 3 | Local Italian Restaurant in Toledo, OH |
| 6-7 | 2 | Olive Garden |
| 7 < | 1 | Deep-dish Pizzeria |

# Project Structure

## Building corpora

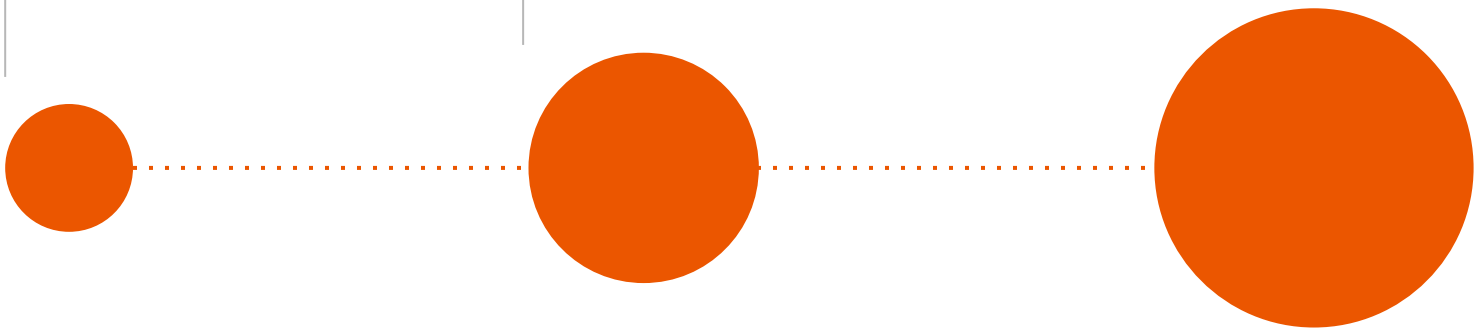
Web scraping and PDF processing

## Parsing data

Matrices and inverted indexes

## Language analysis

Adapt algorithms and models to Italian



# Building Corpora: Data Sources

Name: GialloZafferano
Description: Italian recipe website
Number of recipes retrieved: 55
Corpus: Background Corpus

OpenTable®

Name: OpenTable
Description: online restaurant-reservation service
Number of menus retrieved: 30
Corpus: Background and Test Corpora

Name: Tabula
Description: PDF Scraper
Number of menus retrieved: 25
Corpora: Background and Test Corpora

**Output:** two **json** files containing **dictionaries.**

- **Background Corpus: 90** recipes & menus
- **Test Corpus: 25** menus

# Building Corpora: Code

```
def get_ingredients(url):

    #retrieving the url in json format
    myurl = urllib2.urlopen(url).read()
    soup = BeautifulSoup(myurl)
    ingredients = json.loads(soup.find('script', type='application/ld+json').text)

    #cleaning the text:
    ingredients = str(ingredients)
    ingredients = ingredients.split('[')
    ingredients = ingredients[1:3]
    ingredients = [i.split(']', 1)[0] for i in ingredients]

    ingredients = listToString(ingredients)

    ingredients = re.sub('[^a-zA-ZÀ-ÿ.\s]', '', ingredients)

    ingredients = ingredients.lower()

    return ingredients
```

```
def get_menus(url):
    myfile = requests.get(url)
    soup = BeautifulSoup(myfile.content, 'lxml')
    dishes = soup.select('p[class^="menu"]')

    dishes_cleaned = []

    for dish in dishes:
        dish = str(dish)
        dish = dish.replace('<p class="menu-description__2HXkC4oE">', "")
        dish = dish.replace('</p>', "")
        dish = dish.replace('<p class="menu-section-description__1NOsGasf">', "")
        dishes_cleaned.append(dish)

    dishes_cleaned = [' '.join(i for i in dishes_cleaned)]

    dishes_cleaned = listToString(dishes_cleaned)

    dishes_cleaned = re.sub('[^a-zA-ZÀ-ÿ.\s]', '', dishes_cleaned)
    dishes_cleaned = dishes_cleaned.lower()

    return dishes_cleaned
```

## Parsing data

# How can we look at these corpora and gain insights for our scoring?

# MATRICES!

TF matrix

```
words_freq = {}
for key, value in Background_Corpora_Json.items():
    tokens = get_tokens_stopw(value)
    for token in tokens:
        if token not in words_freq.keys():
            words_freq[token] = 1
        else:
            words_freq[token] += 1
```

TF matrices per doc

```
doc_terms = {}
for key, value in Background_Corpora_Json.items():
    term_freq = {}
    tokens = get_tokens_stopw(value)
    for token in tokens:
        if token not in term_freq.keys():
            term_freq[token] = 1
        else:
            term_freq[token] += 1
    doc_terms[key] = term_freq
```

Inverted Index

```
term_docs = {}

all_tokens = []

for key,value in Background_Corpora_Json.items():
    tokens = get_tokens_stopw(value)
    for tok in tokens:
        if tok not in all_tokens:
            all_tokens.append(tok)
        else:
            continue

for tok in all_tokens:
    doc_dic = {}
    for key,value in Background_Corpora_Json.items():
        tokens = get_tokens_stopw(value)
        if tok in tokens:
            term_docs[tok] = key
        else:
            continue
```

# Dictionary of n-grams per each term

Check out an example here!

| | | | |
|---|---|---|---|
| **spaghetti** | *(spaghetti', 'guanciale')* | *('spaghetti', 'carbonara')* | *('spaghetti', 'cuoceteli')* |
| **spaghetti** | *['gli','spaghetti','alla']* | *[‘cotti’, ‘gli’, spaghetti’]* | *[‘degli’, ‘spaghetti’,‘a’]* |
| **...** | *...* | *...* | *...* |
| **pecorino** | *('pecorino', 'romano')* | *('mantecato', 'pecorino')* | *('crema', 'pecorino')* |
| **pecorino** | *['anche', 'il', 'pecorino']* | *['parte', 'del', 'pecorino']* | *[‘condimento’, ‘al’,’pecorino’]* |

```
grams_dic = {}

for tok in all_tokens:
    grams = []
    for gram in all_ngrams:
        if tok in gram:
            grams.append(gram)

    grams_dic[tok] = grams
```

# Language analysis

Using indexes and matrices, we can apply standard algorithms:

- Edit distance
- Similarity
- SymSpell
- Vectorization
- Classification

Libraries:

- NLTK
- SpaCy
- Stanza
- Gensim
- Stanford CoreNLP
- FuzzyWuzzy

# Could it really be that easy?

Of course not...

# Challenges we've encountered so far:

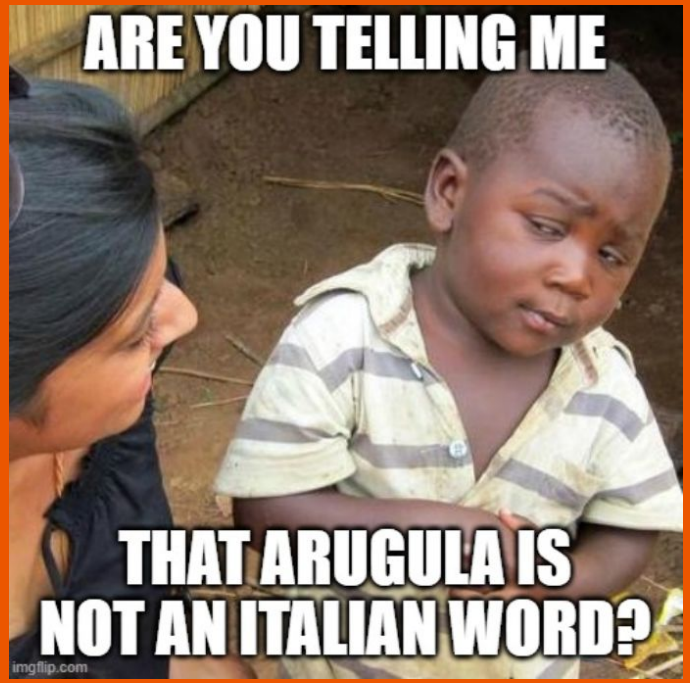
(Hopefully there won't be any more adding to this list)

- Lack of current relevant research
- Complexity of problem (multilingual projects are in general, more of a niche)
- Acquiring Italian data related data to build a background corpus from
- Complicated word combinations that can't be discounted as misspellings
- Finding suitable algorithms that can be fine tuned enough to suit our own purposes

# Ways we could expand the scope of the project:

(Once we get through this semester)

- Expand scope from detecting misspellings to studying cultural assimilation
- Plot regional variations in food item names
- Account for detecting discrepancies in gender agreements
- Classify different types of mistakes that occur frequently and form conclusions


ARE YOU TELLING ME
THAT ARUGULA IS NOT AN ITALIAN WORD?
imgflip.com


WHEN YOU FIND OUT THAT THE "AUTHENTIC" ITALIAN RESTAURANT YOU LOVE
IS NOT AUTHENTIC
THANKS TO FIRE
imgflip.com

# Questions?