GRAS: Generating Recipes with an Algorithmic Sensibility

Théo Faure, theo.faure@insa-lyon.fr
09/2016

***Abstract*— GRAS es una programa de generación de recetas basado en técnicas de aprendizaje de máquina. En este momento, desde algunos ingredientes de entrada, el algoritmo puede generar un titulo de receta que describe la receta y un conjunto de ingredientes que fueron identificados como buenos juntos.**

I. INTRODUCCIÓN

Aprendizaje de máquina está dividido en dos grandes áreas. Los modelos discriminatorios que son más conocidos y más estudiados en la literatura científica, que pueden clasificar datos entre diferentes grupos de manera supervisada o no-supervisada. Pero hay también los modelos generativos (en que se posicione GRAS), que son modelos no-supervisados para generar datos que deben pertenecer a un grupo dado. Estos modelos son más conocidos para generación de texto (robots que hablan solos) y imágenes, pero no hay mucha literatura sobre generación de datos desde características.

GRAS se posicione en está ultima categoría, porque genera recetas desde un conjunto de muchas recetas donde estamos intentando sacar las características que hacen una buena receta.

El proceso empieza con algunos ingredientes de entrada y si uno quiere, tipos de comida (entrada, postre, plato principal, etc). Desde estos elementos, GRAS va a proponer un conjunto de ingredientes para hacer una buena receta y va a poner un título en esta receta. Falta la función de generación de instrucciones.

GRAS usa un modelo generativo basado en probabilidades para inventar las recetas.

II. RECUPERACIÓN DE LOS DATOS

El problema es que en Internet, no hay conjuntos de datos abiertos de recetas de cocina. Para todos uno debe pagar, lo que no quería.

Entonces, hice un trabajo de *web scrapping* en el sitio web marmiton.org, la banca de recetas más conocida en Francia. El web scrapping consista en navegar en un sitio web desde una programa, descargar cada página web y recoger solamente el información que uno quiere con expresiones XPath.

Con eso pude descargar las 60,000 recetas de cocina del sitio web marmiton.org. Descargue también la lista de 2046 ingredientes estándar que tiene el sitió web. Casi todos los ingredientes de las recetas (más de 95 % vienen de esta lista). En en cuadrado siguiente hay un ejemplo de la estructura del json recuperado por este scrapping.

```
{
    "title": ["..."],
    "isVegetarian": true,
    "dishType": "...",
    "link": "...",
    "ingredients": [
        "...",
        "...",
        "..."
     ],
    "instructions": [
        "...",
        "...",
        "..."
    ]
}
```

III. PREPROCESAMIENTO

Estos datos son bien completas, pero no muy bien ordenadas. Los títulos, ingredientes y instrucciones son escritas por la gente, entonces se pueden encontrar faltas de ortografía, los mismos ingredientes escritos de manera diferente, o otro. Entonces, un preprocesamiento fue necesario.

Aquí está la lista del procesamiento hecho en los datos:

1. Poner todo en "lower case".
2. Quitar los acentos franceses.
3. En los ingredientes: dividir entre cantidad, unidad y ingrediente.

Pero todavía hay problemas en el nombre de los ingredientes. A veces hay ingredientes en plural, o en singular, pero se refieren al mismo ingrediente. Entonces, hicimos un index de los ingredientes con un stemmer.

Un stemmer (como el SnowballStemmer de la librería de nltk de python que utilice) es una función que quita todas las marcas de plural o femenino de las palabras, para que la misma palabra en cada forma tenga el mismo stem.

Con eso hicimos un diccionario que asocia un stem a su ingrediente. Con eso, sustituimos todos los ingredientes de la base de datos con ingredientes estándar. La figura III resume el principio del index de ingredientes.

IV. GENERACIÓN DE INGREDIENTES

La primera funcionalidad del programa es de generar los ingredientes de la receta.

IV-A. Datos de entrada

Para generar estos ingredientes, empezamos con datos de entrada. Hay 2 tipos de datos de entrada:

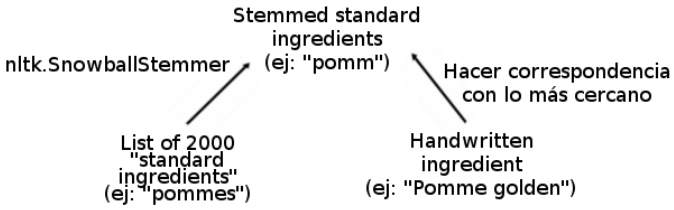


Fig. 1. Index sobre los stem

- Unos ingredientes. Necesita al menos un ingrediente para empezar la generación de una receta. Pero el usuario puede definir un gran número de ingredientes si lo quiere.
- Unos tipos de comida (entrada, postre, plato principal, bebida,...). Eso es opcional: si no hay tipo de comida definido, el algoritmo va a buscar en todos los tipos.

Los argumentos se escriben así en el programa:

```
python generate_recipe.py -i "<ingred1>,
<ingred2>,<ingredN>" -c "<tipo_comida1>,
<tipo_comida2>,<tipo_comidaM>"
```

IV-B. Número de ingredientes

Hay una pregunta, que es el número de ingredientes que el algoritmo debe generar. Había varias maneras de manejar eso, hice una cosa simple que es de usar la media de la cantidad de ingredientes que tienen las recetas que contienen los ingredientes de entrada.

IV-C. Modelo para generar los ingredientes

Queremos generar ingredientes teniendo en cuenta los ingredientes que ya hay en la receta, y la recetas del conjunto de datos de se supongan estar ricas. Con estos datos, hay varios métodos para generar ingredientes.

La técnica que elegimos es un modelo probabilista basado en la teoría de Bayes.

Si I el conjunto de ingredientes ya presentes en la receta, i_{new} el nuevo ingrediente y T los tipos de comidas dados. La formula que vamos a utilizar para generar los ingredientes es:

$$P(i_{new}|I,T) = \frac{P(i_{new} \cap I \cap T)}{P(I \cap T)}$$

Había una otra técnica que parece muy interesante para este tipo de problema que es los "Generative Adversarial Networks"[1]. Es una técnica utilizada para generar imágenes, y el principio es de tener un modelo generativo (G) y un modelo discriminatorio (D) que aprenden al mismo tiempo. El objetivo de G es de generar datos similares a los datos del conjunto de datos, y el objetivo de D es de clasificar los datos entre las que vienen del conjunto et las son generadas. Es una técnica que me gustaría mucho desarrollar para este proyecto porque pienso que podría funcionar, pero es un poco complicado y no tenía el tiempo para eso.

IV-D. Validación de los resultados

Con la técnica de generación utilizada (probabilista), no hay verificación que el conjunto de ingredientes generado está bien.

Hay muchas maneras para validar el resultado, y desarrolle uno de ellos.

- Entrenar un clasificador para saber si un registro viene de los datos o es generado (técnica inspirada de los Adversarial Networks). Eso es la técnica que no desarrolle porque es muy pesada.
- Calcular el mínimo Root Mean Squared Error.[en]tre el conjunto de ingredientes generado y cada receta en los datos. Intentamos tener una RMSE mínima muy cerca de 0, sin tener 0 (sino significa que la receta ya existe en los datos!). Esta técnica esta desarrollada y vemos el resultado en el ejemplo abajo.

V. Generación de título de receta

"Tarte fagilesagne à la à la sucre et chocolat"

"Torta fagilasaña con la con la azucar y chocolate"

La secunda etapa es de generar el título. Ahora que tenemos la lista completa de los ingredientes de la receta, podemos encontrar un título que tiene un sentido en relación con los ingredientes.

Intentemos hacer esta generación de titulo con una Red Neuronal LSTM (Long-short Term Memory) [2] que es una red neuronal recursiva muy conocida para generar texto. Utilizamos el implementación de la librería Keras de Python.

En la figura V vemos el resumen de las capas de la red neuronal.

Layer (type)	Output Shape	Param #	Connected to
lstm_1 (LSTM)	(None, 128)	90112	lstm_input_1[0][0]
dense_1 (Dense)	(None, 47)	6063	lstm_1[0][0]
activation_1 (Activation)	(None, 47)	0	dense_1[0][0]

Total params: 96175

El entrenamiento de la red se hace en los títulos de las recetas del conjunto de datos. Vemos en la figura V la curva de aprendizaje de la red neuronal.

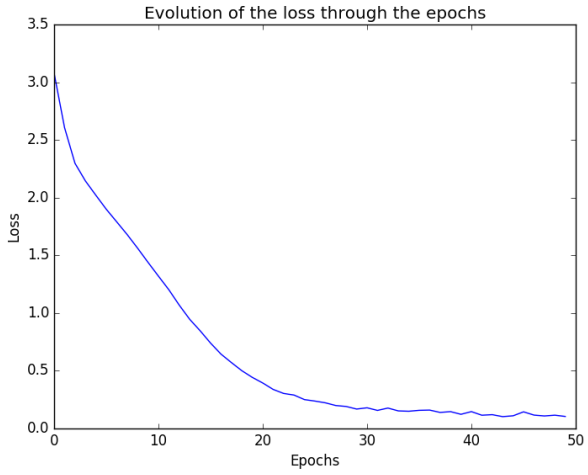
La generación del título se hace con los ingredientos de entrada de la receta. Y después, cogemos la frase (entre dos puntos) que corresponde lo mejor al conjunto de ingredientes (buscando si hay los nombres de los ingredientes en el título.

VI. Resultados

Intentamos el programa con un ejemplo bien representativo del funcionamiento del programa. Intentamos generar un plato principal con los ingredientes Resz .Arina".

```
$ ./gras -i "Boeuf,Farine" -c
"PlatPrincipal"

IN: ["Boeuf", "Farine"] ["PlatPrincipal"]
```



OUT: Boeuf, Farine, Beurre, Bouillon, Tomate, Pinot noir, Madere, Bouquet garni, Pomme, Oignon, Carotte, Ail

Title: Tere aux pommes lr sercin migrais de a la noix de coco faton vin gres le nois cocrons brine carrt boeuf, farine, beurre et bouillon.

Validation error: 0.0221

- La lista de los ingredientes es coherente y un poco original: hay una combinación de carne y frutas, de vino rojo y blanco. No es muy común pero porque no, se puede intentar!
- Sin embargo el titulo de la receta no tiene ningún sentido. No tiene relación con la lista de ingredientes y hay muchos palabras inventadas. Intentemos con muchos ingredientes, y hay pocas veces cuando el título realmente tiene un sentido. Quizás se podría corregir cambiando la función de perdida en el aprendizaje del modelo, para añadir la coacción de coherencia con los ingredientes.

VII. Conclusiones

- Los modelos generativos forman un área de Aprendizaje de Máquina muy interesante pero muy diferente de los modelos discriminatorios. Se piense al revés: *no es clasificar un registro, pero crear un registro bien clasificado*.
- Hay menos literatura sobre modelos generativos, y menos técnicas conocida. Entonces son problemas difíciles que necesitarían mucho tiempo, porque siempre se pueden mejorar.
- Es un problema que queda mucho al apreciación del desarrollador porque no hay un objetivo claro sino "generar buenas recetas". Este aspecto me gusta mucho, porque es motivo de reflexión y da espacio para creatividad.

VIII. Trabajos futuros

Hay todavía muchas cosas que se pueden mejorar en este proyecto, aquí está una corta lista.

- Podríamos usar al técnica de Adversarial Networks para generar los ingredientes.
- La función de generación de instrucciones no esta hecha, y debería ser para tener un real generador de recetas. Pienso que es un trabajo bastante difícil porque realmente necesita tener un sentido en relación con los ingredientes, pero también se necesita una relación en la secuencia de los instrucciones.
- Pienso que se podrís mejor el desempeño y de pronto la calidad de los resultados utilizando Principal Component Analysis (PCA) para reducir la dimensión de los datos. Intenté un poco en este proyecto utilizar PCA pero sin éxito.
- Hay una competición de Kaggle que da un conjunto de datos para clasificar recetas entre países. Sería una muy buena funcionalidad poder preguntar una comida de una nacionalidad especial.

References

[1] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets, in: Advances in Neural Information Processing Systems, 2014, pp. 2672–2680.

[2] F. Gers, Long short-term memory in recurrent neural networks, Ph.D. thesis, Universität Hannover (2001).