TP 1 : Kaldi, premier contact

École thématique CNRS – Big Data Speech

Roscoff, Juillet 2018

L'objectif de cette séance de TP est de prendre en main Kaldi, une boîte à outils consacrée à la reconnaissance automatique de la parole.

Notre objectif sur les deux séances : construire des modèles acoustiques de type HMM/GMM et être capable de réaliser un alignement forcé mot ou phonème/signal.

Autour de Kaldi s'est construite une communauté de développeurs et d'utilisateurs très importante, qui permet à ce projet *open source* de suivre l'évolution constante et souvent rapide de l'état de l'art du domaine. Il s'agit toutefois d'une boîte à outils complexe, qui n'est pas facile à prendre en main pour les non spécialistes.

Cette séance de TP offre un premier contact avec Kaldi pour des tâches bien précises, en particulier la préparation des données pour l'apprentissage de modèles acoustiques et l'extraction de paramètres acoustiques.

1 Configuration de Kaldi

Une première étape consiste à construire son propre environnement de travail. Pour cela, rendez-vous d'abord à la racine de votre compte (par défaut vous devriez y être dès l'ouverture du terminal, mais cela n'est pas toujours vrai), en tapant la commande :

```
cd ~
```

Puis créez votre répertoire de travail :

```
mkdir tp_kaldi
```

Avant de positionner le shell dedans :

```
cd tp_kaldi
```

Pour faciliter son utilisation, Kaldi offre un ensemble de 'recettes' (*recipes*') qui contiennent un ensemble de scripts à utiliser tel quel pour construire des systèmes de reconnaissance de la parole spécifiques à un corpus donné. Nous pouvons nous inspirer de ces recettes pour construire notre propre système.

Dans un premier temps, nous allons copier quelques fichiers d'une recette déjà existante (celle du corpus TEDLIUM) dans notre répertoire de travail, et créer deux nouveaux sous-répertoires.

```
cp -r /opt/kaldi/egs/wsj/s5/steps .
cp -r /opt/kaldi/egs/tedlium/s5/utils .
cp -r /opt/kaldi/egs/tedlium/s5/local .
cp -r /opt/kaldi/egs/tedlium/s5/path.sh .
cp -r /opt/kaldi/egs/tedlium/s5/conf .
cp -r /opt/kaldi/egs/tedlium/s5/run.sh .

mkdir data
mkdir exp
```

steps est un répertoire qui contient des scripts pour créer un système de reconnaissance de la parole (étape par étape). Il s'agit de scripts pour l'apprentissage des modèles, mais aussi le décodage.

utils contient des scripts pour modifier certains fichiers de Kaldi, par exemple pour découper les répertoires de données en lots plus petits

local contient des fichiers spécifiques au corpus sur lequel nous travaillons (ici il s'agissait du corpus TEDLIUM).

data contiendra des répertoires de données. Nous en créerons bientôt...

exp contient les expériences courantes, les modèles créés, ainsi que les fichiers de log.

conf contient les fichiers de configuration nécessaires pour certains scripts.

path.sh contient le chemin vers le répertoire d'origine de Kaldi.

run.sh est une recette, c'est-à-dire un script qu'il suffit d'exécuter pour construire un système de reconnaissance de la parole. Ce type de script permet de comprendre comment utiliser les différents outils fournit par Kaldi.

Les fichiers exécutables de Kaldi (c'est-à-dire les logiciels prêts à l'emploi après installation complète de Kaldi) sont stockés dans un autre répertoire que celui que nous avons créé. L'idée est que n'importe quelle expérience puisse utiliser les mêmes exécutables, sans avoir besoin de les dupliquer. Pour utiliser ces logiciels à partir de n'importe où, il est nécessaire :

1. de modifier la variable d'environnement KALDI_ROOT à l'intérieur du fichier *path.sh* (on ne le fera qu'une seule fois). Avec votre éditeur de texte préféré (par exemple *gedit*, en mode texte, plus simple d'utilisation que *vi*, ou bien n'importe quel éditeur en mode graphique), modifiez comme ceci la ligne correspondante du fichier *path.sh* :

```
export KALDI_ROOT=/opt/kaldi
```

2. et de faire connaître cette valeur (il s'agit du répertoire dans lequel Kaldi a été installé, c'est le chemin 'racine' de Kaldi) au shell en tapant (à chaque nouvelle session du shell pour laquelle on voudra utiliser Kaldi) :

```
source path.sh
```

ou

```
./path.sh
```

Pour vérifier que la variable d'environnement a correctement été prise en compte, tapez

```
echo $KALDI_ROOT
```

Cette commande doit afficher le chemin 'racine' de Kaldi.

2 Préparation des données

Pour construire son système Kaldi pour son propre projet, il est conseillé de s'inspirer d'une recette déjà existante. Dans cette séance, nous nous inspirons de la recette construite pour TEDLIUM, que nous allons adapter à notre corpus.

Quel que soit le format des données initiales, il est nécessaire de les présenter sous la forme attendue par Kaldi. À partir de là, les outils et recettes proposés par Kaldi seront beaucoup plus faciles à utiliser.

Tout d'abord, il faut récupérer les données, à partir d'un disque local ou d'un serveur distant. Pour ce TP, les données ont déjà été stockées localement. Afin de ne pas encombrer le disque dur en dupliquant des données qui ne seront utilisées qu'en lecture, il est préférable de créer un lien symbolique vers le répertoire contenant ces données :

```
ln -s /home/fr2424/stage/stage01/BDS/TP_Kaldi/db ~/
     tp_kaldi/db
```

1. Examinez le contenu du répertoire *db*, sa structure, et les fichiers qu'il contient. La commande *more* est particulièrement utile :

```
more db/train/stm/20030428_1400_1500_RFI_ELDA.stm
```

2. Copier le répertoire suivant, qui correspond au répertoire local d'une recette Kaldi, adaptée à cette séance de TP.

```
cp -r /home/fr2424/stage/stage01/BDS/TP_Kaldi/
     local_BDS .
```

3. Consultez le contenu du script *local_BDS/prepare_data_BDS2018.sh* avec un éditeur de texte, par exemple *gedit* :

```
gedit local_BDS/prepare_data_BDS2018.sh
```

4. Exécutez-le :

```
local_BDS/prepare_data_BDS2018.sh
```

5. Explorez le répertoire *data*. En particulier, attardez-vous sur les fichiers *text*, *segments*, *utt2spk*, *spk2utt*, *wav.scp* et *reco2file_and_channel*. Ce sont des fichiers essentiels pour Kaldi. La figure [1] suivante résume la structure et le contenu du répertoire contenant ces fichiers :

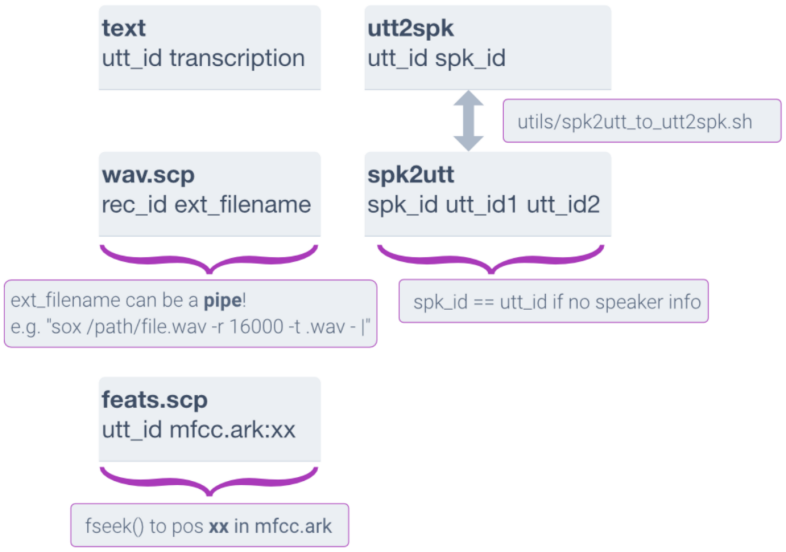

FIGURE 1 – Structure et contenu d'un répertoire de données Kaldi

6. Validez chacun des répertoires crées dans le répertoire data :

```
utils/validate_data_dir.sh --no-feats data/train.orig
utils/validate_data_dir.sh --no-feats data/test.orig
```

3 Calculs de paramètres acoustiques MFCC

Nous allons maintenant produire des paramètres acoustiques qui seront ensuite utilisées par des modèles acoustiques de type GMM/HMM. Les paramètres acoustiques les plus courants pour ces modèles sont des paramètres de type MFCC ou PLP.

En nous plaçant dans le répertoire data :

1. Cette figure provient de `https://www.inf.ed.ac.uk/teaching/courses/asr/2017-18/lab1.pdf`, un TP de l'université d'Edinbourg qui a inspiré cette séance de TP

```
cd ~/tp_kaldi/data
```

utilisons le script *make_mfcc.sh* qui se trouve dans le répertoire **steps** (n'hésitez pas à jeter regarder le contenu du script *make_mfcc.sh* avec un éditeur de texte).

```
for dir in train test; do
  steps/make_mfcc.sh data/$dir.orig exp/make_mfcc/$dir
       mfcc
done
```

Cette commande bash permet d'exécuter plusieurs fois la même commande (il s'agit d'une boucle). La variable *$dir* aura pour valeur *train* lors de la première exécution de *make_mfcc.sh* et *test* lors de la seconde.

Une fois réalisée, cette commande aura créé le fichier *feats.scp* dans un répertoire *mfcc* dans les sous-répertoires *train* et *test*, avec l'archive correspondante (*feats.ark*, nous allons bientôt y venir).

Exercice

1. Recommencez l'étape précédente (la boucle *for* avec la commande *make_mfcc.sh*) en y intégrant le calcul des statistiques nécessaires pour la normalisation de la moyenne et de la variance cepstrale. Inspirez-vous du script *run.sh*.
2. Validez de nouveau le répertoire.