Université de Montréal

IFT6285 (TALN) — Devoir1
Modèles n-grammes discrets avec `kenlm`

Contact :
Philippe Langlais
RALI/DIRO
Université de Montréal

+1 514 343 61 11 ext: 47494
felipe@iro.umontreal.ca
http://www.iro.umontreal.ca/~felipe/

dernière compilation : 21 septembre 2023 (13:55)

Ressources

Vous avez accès à un corpus populaire pour entraîner et tester des modèles de langue : le 1B-word corpus (1Bwc) (Chelba et al. 2013). Il s'agit d'un corpus assez volumineux que j'ai installé au DIRO (soit `[rep]` ce répertoire) et que vous pouvez consulter en lecture sans avoir à le télécharger [1] :

`/u/demorali/corpora/1g-word-lm-benchmark-r13output/`.

À l'aide du package `kenlm`, vous allez entraîner des modèles de type (modified) Kneser&Ney (Chen et Goodman 1996) de manière efficace en temps et en mémoire. Les détails de ce package sont décrits dans (Heafield 2011). Une API python (à installer) vous permet de manipuler simplement un modèle déjà entraîné. Vous entraînerez vos modèles sur un sous-ensemble des 99 tranches du répertoire :

`[rep]/training-monolingual.tokenized.shuffled/`

Une tranche est un fichier (e.g. `news.en-00001-of-00100`) constitué d'environ 305k phrases préalablement découpées en mots (l'espace est séparateur de mots). Les tests sont à conduire sur la tranche `news.en-00000-of-00100` dont une copie est disponible ici :

`[rep]/heldout-monolingual.tokenized.shuffled/`

Bien que je vous encourage à exécuter vos programmes sur une machine du DIRO [2] (en utilisant le corpus installé sur `/u/demorali`), vous pouvez télécharger le corpus depuis ce github. Il s'agit d'une ressource de 11Go (soyez assurés d'avoir une bonne connexion internet). J'ai également préparé 1bshort.tar.gz qui contient les 9 premières tranches de ce répertoire (153Mo), ainsi que la tranche de test (18Mo). Vous pouvez ne considérer que les 1000 premières lignes de la tranche de test.

À faire

1. Installer le package `kenlm` sur votre machine. Cela impose d'avoir CMake d'installé. Alternativemenent, `kenlm` est installé au DIRO dans : `/u/demorali/bin/x86_64/moses_3.0/bin/` (les commandes `lmplz` et `/build_binary` sont présentes).

1. Du moins si votre programme est lancé sur une machine du DIRO...
2. Attention ! Il n'est pas autorisé d'utiliser le serveur `arcade` pour lancer vos calculs.

2. Écrire un rapport au format pdf nommé `rapport-devoir1-noms` où `noms` est à remplacer par votre (ou vos) nom(s) et contenant les informations suivantes :

   (a) La moyenne, le min et le max de perplexité[3] obtenus sur les 1000 premières phrases de la tranche de test par un modèle bigramme entraîné sur les 9 premières tranches du répertoire d'entraînement. Vous préciserez quelle mesure de perplexité vous utilisez.

   (b) Préparez une (ou plusieurs) figure(s) montrant l'impact du nombre de tranches d'entraînement considérées sur la performance du modèle en test, sur sa taille sur disque et sur le temps d'entraînement. Accompagnez ces figures d'une analyse sommaire. Vous prendrez soin d'identifier l'architecture matérielle sur laquelle vous exécutez votre devoir et les caractéristiques des modèles étudiés (ordre, prétraitement, etc.).Votre réponse à cette question doit tenir en au plus une page (incluant les figures).

   (c) Proposez votre "meilleur modèle", en vous basant sur vos expériences et indiquez sa perplexité sur le corpus des 1000 premières phrases du corpus de test. Vous indiquerez brièvement (au plus un paragraphe) les caractéristiques de ce modèle.

   (d) Pour les questions qui suivent, indiquez la réponse et rapportez le temps qu'il a fallu pour votre programme à retourner la réponse. Si votre programme a été écrit à l'aide d'une ligne de commande shell (csh ou autre) mentionnez la, sinon indiquez le nombre de lignes et le language de votre solution.

      i. Combien y a t-il de lignes (phrases) dans `1bshort` ?

      ii. Existe t-il dans `1bshort` des phrases vues au moins 3 fois ? Si oui lesquelles ? Au moins 4 fois ?

      iii. Combien de phrases dans `1bshort` ont exactement 20 caractères ? 120 caractères ?

      iv. Quelle sont les longueurs minimum et maximum (en nombre de caractères) des phrases de `1bshort` ?

      v. Quelles sont les 100 phrases de `1bshort` les plus fréquentes qui ont moins de 21 caractères ?

3. Vous pouvez par exemple utiliser la fonction `perplexity` du package python.

vi. Quels sont les 6 mots les plus fréquents après `continue to` dans `1bshort` ? On appellera mot une chaine sans espace, sans ponctuation et sans chiffre. On ne fera pas de différence entre les majuscules et les minuscules (`En` et `en` sont des occurrences du même type).

vii. Quelle est la séquence de 10 mots la plus fréquente en début de phrase dans `1bshort` ? On considérera un mot comme une chaine séparée par un espace (une ponctuation est un mot).

viii. Quels sont les mots d'exactement 4 lettres majuscules les plus fréquentes dans `1bshort` ?

Repère

Sur les 9 tranches du corpus 1bshort.tar.gz discuté plus haut, entraîner un modèle bigramme sur un iMac à 3.6 Ghz Intel Core i7 (16 Go de mémoire), requiert 22 secondes. Le modèle résultant prend sur disque 239Mo (format arpa), ou 181Mo (format binaire). Entraîner un modèle trigramme prend le double de temps et requiert la bagatelle de 1.2Go (arpa) ou 761Mo (binaire).

Il ne vous sera donc peut-être pas possible d'entraîner un modèle sur les 99 tranches disponibles. Ça n'est pas grave !

`kenlm` et Python

Python est le language le plus facile à utiliser pour interroger un modèle déjà entraîné par `kenlm`. Voir ici (en bas de page) pour l'installation, des exemples ansi qu'une documentation à même le code...

Remise

La remise est à faire sur Studium sous le libellé `devoir1`. Vous devez remettre votre code et votre rapport (format pdf, texte en anglais ou en français) dans une archive (gzip, tar, tar.gz) dont le nom est préfixé de `devoir1-name1` ou `devoir1-name1-name2` selon que vous remettez seul ou a deux, où `name1` et `name2` sont à remplacer par l'identité des personnes faisant

la remise (`prénom_nom`). Donc si j'avais à remettre seul mon solutionnaire au devoir1, je le ferais sous le nom `devoir1-philippe_langlais.tar.gz`. Assurez vous que le nom des personnes impliquées dans le devoir soit indiqué sur tous les documents remis (code et rapport). Le devoir est à remettre en groupe d'au plus deux personnes au plus tard lundi 2 octobre à 23h59.

Note : Aucune donnée ni modèle n'est demandé : juste le rapport et le code.

Chelba, Ciprian et al. (2013). "One Billion Word Benchmark for Measuring Progress in Statistical Language Modeling". In : *CoRR* abs/1312.3005. URL : http://arxiv.org/abs/1312.3005.

Chen, Stanley F. et Joshua Goodman (juin 1996). "An Empirical Study of Smoothing Techniques for Language Modeling". In : *34th Annual Meeting of the Association for Computational Linguistics*. Santa Cruz, California, USA : Association for Computational Linguistics, pages 310-318. DOI : 10.3115/981863.981904. URL : https://www.aclweb.org/anthology/P96-1041.

Heafield, Kenneth (juill. 2011). "KenLM : Faster and Smaller Language Model Queries". In : *Proceedings of the Sixth Workshop on Statistical Machine Translation*. Edinburgh, Scotland : Association for Computational Linguistics, pages 187-197. URL : https://www.aclweb.org/anthology/W11-2123.