Université de Montréal

IFT6285 (TALN) — Devoir2
Modèles ngrammes 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 : 17 septembre 2020 (08:29)

Données

Dans ce devoir, vous allez manipuler le package `kenlm` qui permet d'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 allez entraîner des modèles sur le même corpus que celui que vous avez manipulé lors du devoir 1 : le 1B-word corpus (Chelba et al. 2013) dont une copie est disponible au DIRO (soit `[rep]` le répertoire associé) :

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

Les entraînements sont à faire sur un sous-ensemble des 99 tranches du répertoire `[rep]/training-monolingual.tokenized.shuffled` et 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 (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).
2. Indiquez dans votre rapport (format pdf) la moyenne, le min et le max de perplexité[1] 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.

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

3. Ajoutez dans votre rapport une ou plusieurs figures 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 d'au plus une page. Vous prendrez soin d'identifier l'architecture matérielle sur laquelle vous exécutez votre devoir.
4. Analysez sur au plus une page supplémentaire le meilleur modèle que vous avez entraîné. Laissez aller votre curiosité. Au minimum, indiquez la perplexité moyenne des phrases de test.

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 `devoir2`. 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 `devoir2-name1` ou `devoir2-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

devoir2, je le ferais sous le nom `devoir2-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 vendredi 25 septembre 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 (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 (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.