**Instructions for Telegram bot hosted on AWS [Optional]**

This part of the project is completely optional and it is not graded towards a certificate in this course. It sets a fun and ambitious goal of creating an interactive assistant in a real messenger (Telegram). Therefore, apart from NLP skills it will require some technical skill to set up the environment. The reading below contains step-by-step instructions, but we know it can still be tough. A good part though is that in real life / career you will likely need those skills too, so it's not a waste.

**Here are some links at a glance related to the technical part of the project:**

* [Starter code for the Telegram bot](https://github.com/hse-aml/natural-language-processing/blob/master/optional/telegram_bot)
* [Guide on setting up AWS machine](https://github.com/hse-aml/natural-language-processing/blob/master/AWS-tutorial.md)
* [Docker guide](https://github.com/hse-aml/natural-language-processing/blob/master/Docker-tutorial.md)

Docker guide also has a FAQ and an instruction on reporting your issues to Coursera forum in a way that helps us to debug your issue.

**On alternative technologies**

Our instructions guide you on setting up the project with Telegram and AWS. However you are also welcome to try any other stack of technologies. In case you have a server available, you can use it instead of the Amazon machine. In case you are facing difficulties with registering Telegram account, one particular alternative way would be Flask. There is a [5-minute demo](https://github.com/chamkank/flask-chatterbot) on how to set it up and get a web-server that servers as a bot.

**Part 1. How to host a bot on a remote server (AWS).**

You will need to do some setup steps, please, closely follow the instructions below.

* We suggest that you create a free tier instance with Amazon Web Service (AWS). You can find a [step-by-step instruction on our github](https://github.com/hse-aml/natural-language-processing/blob/master/AWS-tutorial.md). Upon completing it, you will have an instance. Then you can run the Docker on it ([see instruction for Ubuntu](https://github.com/hse-aml/natural-language-processing/blob/master/Docker-tutorial.md)) with all required dependencies set up.
* After you ssh to the remote machine, make sure to work inside a *tmux session.* This will ensure that your session (and the running bot) will be kept alive even even if you lose your connection. Thus the usual order would be: ssh -> tmux -> docker -> python.

All you need to know about tmux is: *(pic here)*

But feel free to look at a more [detailed tutorial](https://robots.thoughtbot.com/a-tmux-crash-course) and incorporate tmux into you day-to-day life.

**Part 2. How to set up 'Hello world' Telegram bot.**

We are going to integrate the bot to [Telegram](https://telegram.org/) messenger. To do so, you will need to create a *token* and use it to run the bot.

* Talk to @BotFather in Telegram. The command "/newbot" will create a bot for you. You will be prompted to enter a name and a username for your bot. After that, you will be given a token.
* Run the 'Hello world' bot with the script provided for you: *python3 main\_bot.py --token=YOUR\_TOKEN*

You can now talk to this bot in Telegram (make sure to talk via the messenger, not via your console).

In case you are interested to know more about how it works, you can have a look into these tutorials: [[1],](https://core.telegram.org/bots#creating-a-new-bot) [[2].](https://djangostars.com/blog/how-to-create-and-deploy-a-telegram-bot/)

**Part 3. Putting everything together.**

Now follow instructions in *main\_bot.py* to extend the bot with Stackoverflow and chit-chat functionality, that you’ve developed earlier in this course. We suggest you first do this locally and check that you can successfully talk to your bot via Telegram.

Next, move all your scripts, data, and pre-trained models to AWS and mount the directory such that it is visible from the Docker. Run the bot and test it again.

This assignment is not graded. Feel free to share the link with your friends or on social media - it’s a nice accomplishment! :)

**Troubleshooting AWS memory errors:**

Here is some advice in case you see memory errors when running your bot on the Amazon machine.

* Make sure that you work through the IPython notebook and train models locally on your laptop. We suggest that you use 1GB Amazon machine only to serve the running bot (python scripts).
* Check your vocabulary size for word embeddings. Whether you are using your custom StarSpace model or downloaded word2vec/GloVe vectors, the vocabulary should be filtered based on the StackOverflow train corpus. As a rule of thumb, a size of ~100k words is enough for most of NLP tasks you deal with (if you are not Google, of course :).
* Make sure you never keep in RAM question embeddings for all categories. Instead, pick one category using the classification model and read the corresponding embeddings from disk on the fly.
* If the problem still persists, another simple solution might be to take a machine with more RAM (unfortunately, not available at free tier).