


Peer Assessments (https://stanford.coursera.org/cs124-002/human_grading/) / PA6 Machine Translation Help (https://stanford.coursera.org/cs124-002/help/peergrading?url=https%3A%2F%2Fstanford.coursera.org%2Fcs124-002%2Fhuman_grading%2Fview%2Fcourses%2F971938%2Fassessments%2F8%2FpeerGradingSets)

Submission Phase

1. Do assignment ☒ (/cs124-002/human_grading/view/courses/971938/assessments/8/submissions)

due in 4day 17h

Evaluation Phase

2. Evaluate peers ☐ (/cs124-002/human_grading/view/courses/971938/assessments/8/peerGradingSets)
3. Self-evaluate  (/cs124-002/human_grading/view/courses/971938/assessments/8/selfGradingSets)

Results Phase

4. See results  (/cs124-002/human_grading/view/courses/971938/assessments/8/results/mine)

[← Return to list \(/cs124-002/human_grading/view/courses/971938/assessments/8/peerGradingSets/8\)](/cs124-002/human_grading/view/courses/971938/assessments/8/peerGradingSets/8)

[▶ Preview next submission \(/cs124-002/human_grading/view/courses/971938/assessments/8/peerGradingSets/8/peerGradingSets/1\)](/cs124-002/human_grading/view/courses/971938/assessments/8/peerGradingSets/8/peerGradingSets/1)

You are urged to preview the work of all 3 of your peers before you start submitting your required evaluations.

Remember to click the "Submit Evaluation" button after *each* time that you fill out an evaluation.

3 remaining of 3 required evaluations

[Save draft](#)

[Submit evaluation](#)

Submission from: Student 1

Homework 7: Machine Translation

Due: Friday February 28 at 5:00pm

For this assignment, you are going to experiment with the **direct** approach to **Machine Translation** on a small set of data. You will implement the direct system as a baseline, enhance its performance with post-processing, and then produce a critical assessment of your system. For more information, see the FAQ on Piazza.

(NOTE: No starter code for this assignment. See below for submission instructions.)

Language

First your group needs to choose a language. You will be translating **from** this other language ('F') **to** English ('E'). You do not need to be fluent speakers of language F, but at least one person in your group should know it well enough to be able to roughly assess the quality of the translation. If none of you know any language other than English well enough, just choose the one you know best. You might want to ask a friend who speaks the chosen language to help you by pointing out errors in the translation to you. (***Your friend should not design post-processing strategies -- see below -- for you!*** But you can ask about translation errors and potential corrections, and then work on spotting the patterns in those corrections.)

Data

To build your system, you will need a small working corpus on which you can test it. Your first job is to create that corpus. It should be 15 sentences. Don't write the sentences yourself; take real sentences from a source in your chosen language, such as a newspaper, a novel, a web site, etc.. You can have sentences from different sources. All the sources should be included in your write-up.

Dictionary

We're assuming a closed vocabulary system, so you will have access to a dictionary for all the words in the working corpus. (A complex real-world system would work with an open-vocabulary assumption, and deal with new words on the fly.)

Create a bilingual E-F (English / Language-F) dictionary for each word in your working corpus. It's difficult to get a good downloadable dictionary, so do this using a web-based or print English-X dictionary (Here's a [web-based dictionary](http://www.wordreference.com/) (<http://www.wordreference.com/>) for several languages. [Google Translate](http://translate.google.com) (<http://translate.google.com>) often works quite well for word to word translation, too.)

Don't try to work directly with the entire dictionary. Rather, create a little dictionary file that has just the words in your working corpus and the corresponding translation for English. Note that, if you have more than one translation for some word, you can put all the translations in the dictionary, and you will need a heuristic to choose the correct one in context when you are translating a sentence. (That could just be using the most frequent translation, but you can use information about the source sentence, a language model, etc..) ***Handpicking the correct translations in advance is not allowed.***

Dev-test split

From this point on, leave out about 5 sentence pairs for testing, and work only with the rest of the pairs when developing your system. The set of 5 sentences is your test set; the set of 10 sentences is your dev set, which you use both as a source of insights into the translation problem, and to produce evaluations during the development process. At the end, you will come back to the test set to see if the system you developed generalizes well.

Don't look at the test set while you're developing! The reason for this is that, the more you know about your test set, the more this allows you to tailor your system to perform well on that set, and that's not a fair evaluation. It's a violation of the honor code to look at the test set after you built your dictionary (which should be your very first step).

Translation system

Now write code (in Java or Python, as usual) to implement the following **"Direct MT"** system:

(NOTE: There is no starter code for this project, so you are free to design as you like. Unlike previous assignments, the Submit script does not invoke your code, just collects your source and your write-up. See **Submit** section below.)

- Use your bilingual dictionary to translate each word from Language F into English.
- Obtain any annotation you want on your sentences -- word tokenization, lemmatization, POS-tagging, parsing, word sense disambiguation, anything you need. Do not write your own tools for this; you can look for a toolkit in your favorite language, and the staff can help with that too. (Note that no specific type of annotation is required, and you will not be graded on the type of annotation you choose. It is using these tools intelligently that matters.)
- Now write code for 6-10 **pre- or post-processing strategies** to improve the baseline translations from the direct method. Anything that you can do automatically is fair game here: reordering words based on part-of-speech (nouns and adjectives, for instance, as we saw between English and Spanish); substituting trees; reordering constituents; using a language model... the sky is the limit! Good strategies will be ones that generalize well and produce significant improvements on the translation, making it look more like real English. Leverage your knowledge of the languages, and try to spot patterns in your dev set.

Testing

When you're finished with your translation system, run it on the test set. Your code will need to run the direct MT system based on the dictionary you compiled in the beginning, then produce whatever annotations you need for your post-processing strategies, then execute those strategies.

Note that we usually take care of this and run your code on a test set that you haven't had access to. The logistics for this assignment is different: you will run the system on the test set yourself. We count on your honesty for this step: ***do not use the test set while you are developing, and do not go back to developing after you evaluate your system on the test set.***

Error Analysis

After you're all done and have produced your translations for the test set, there will inevitably still be errors. (In fact, you probably still have errors in the dev set as well -- MT is hard!) Now is the time to think deeply about those errors and how to make the system better.

Make sure you leave plenty of time for this: error analysis is one of the most important stages in the development of complex systems! Where are things going wrong: maybe there's ambiguity in the source sentences? Or perhaps idiomatic meanings? How is the fluency of the output? You should not only identify the errors in your output, but also think carefully about what aspects of language make the problem difficult; what simplifying assumptions in your approach fail

to tackle those aspects of language; what information would be needed to avoid the errors; and what might be some ways of getting that information. Machine translation is an open area of research, so of course you are likely to run into errors that are very difficult to avoid. What's important in this assignment is that you understand why those errors happen.

Google it!

Next, run your sentences through Google Translate and discuss any errors that Google makes. Where does Google do better? Are there places where your implementation does better? Why?

Report

Provide a project write-up titled "report.pdf". It should include at least the following:

- A comment on the language F that you chose. You should make a brief statement of particular challenges in translating your choice of F language to English (relative to other possible choices for F), and key insights about the language that you made use of in your post-processing strategies.
- Your corpus of 15 sentences, with clear indication of the dev-test split.
- The output of your system.
- For each post-processing strategy you implement, a description of what differences between Language F and English that strategy was designed to address. Make sure you motivate the strategies by pointing to the characteristics of the dev set that led you to design them.
- Your error analysis, including specific reference to what your code does and ideas for how further work might fix your remaining errors.
- The output of Google Translate.
- A comparative analysis commenting on your system's performance compared to Google Translate's. Show where the systems agree, what your system does better than Google Translate, and what Google Translate does better than your system.

It's very important to write a good report, since everyone will be writing different systems for different languages, so your peer graders will not know much a priori about what you did. You should explain your strategies and comment on your errors clearly and concisely, with examples.

Limit: *roughly* 2000 words. (That's about 4 pages if you were to format it.)

Evaluation of your work

While your code will be submitted and your graders may inspect it they choose to, your score for this assignment will be based on your write-up. The grade is based on at least the following:

- Your **relative** progress towards a good English translation. (To be clear, we're gauging your **progress**, not an absolute measure of how close you get. Depending on your language choice, you might start out farther or closer. There's no grading advantage here in picking a language that starts out closer.)
- Your thoughtful choices in designing general, robust strategies.
- The insightfulness of your error analyses, both of your own MT and for Google Translate.
- Clarity of your write-up.

Be aware of the fact that, if a grader does not understand something in your write-up, they may (generously!) choose to look at your code. In that situation, having clear, documented code will be your chance to convey your ideas and get points for them.

Submit

Unlike previous assignments, your code will not be submitted through our `submit` script. Rather, you will upload a zipped file containing all your code, data and a PDF (only PDFs accepted!) of your write-up to `/afs/ir/class/cs124/PA6Submissions`. The name of your .zip file should be a long, kooky team name that has no information about the team members. (We will make a copy of your file so that you don't appear as the owner.)

Additionally, **email a brief statement of collaboration to cs124-win1314-staff@lists.stanford.edu (<mailto:cs124-win1314-staff@lists.stanford.edu>) with the subject line "PA6 Team"** naming the members of the team and explaining their responsibilities in the project. (Team members will normally get the same grade, but we reserve the right to differentiate in egregious cases.) It's really important that you don't forget this, because there will be no information in your Coursera submission about who you worked with.

IMPORTANT (1): There are no late days.

IMPORTANT (2): Only one student from each group will submit. (It doesn't matter who.) Use the same login for peer grading later.

IMPORTANT (3): YOUR SUBMISSION SHOULD BE COMPLETELY ANONYMOUS; do not include any name information in any of the files you are submitting.

Submit your work! Upload a zipped file containing

- a code/ folder with all your code,
- a corpus/ folder with your working corpus (including clear indication of the dev-test split),
- an output/ folder with your translations (both system and baseline translation) of all 15 sentences in the working corpus,
- a google/ folder with the Google Translate results for all 5 sentences in the test set, and
- a PDF with your write-up.

to /afs/ir/class/cs124/PA6Submissions. Give that file a long, kooky name and **put the name of the file here**.

You don't have to write anything additional, but if you have any notes for the graders about the file you're submitting, this is the place for them.

cs124translate-team4444.zip

[flag \(/stanford.coursera.org/cs124-002/help/flag?url=https%3A%2F%2Fstanford.coursera.org%2Fcs124-002%2Fhuman_grading%2Fview%2Fcourses%2F971938%2Fassessments%2F8%2Fsubmissions%2F58\)](https://stanford.coursera.org/cs124-002/help/flag?url=https%3A%2F%2Fstanford.coursera.org%2Fcs124-002%2Fhuman_grading%2Fview%2Fcourses%2F971938%2Fassessments%2F8%2Fsubmissions%2F58)

Overall evaluation/feedback

Introduction (max. 4 points)

The paper must appropriately introduce the language pair, with a brief comment on key differences between the languages.

- 0 points if there is no discussion of specific linguistic properties of language F;
- 3 points if at least three properties of language F (that make it different from English) are adequately discussed;
- 4 points if more than five properties of language F are adequately discussed, or at least three properties are discussed in some detail (with examples, discussion of exceptions, etc.).

Working corpus (max. 0 points)

The corpus should be constructed according to the instructions of the assignment. -2 points for **each** of the following problems:

- there are not enough sentences in the working corpus;
- the dev/test split is not clear;
- not all sentences have their sources clearly indicated.

Translation system (max. 10 points)

For **each** pre- or post-processing strategy presented, increment the number of points x by:

- 0 points for a strategy that does not apply **in the dev set**. The authors are responsible for making it clear where the rule should apply.
- 0 points for a strategy that applies only to one sentence **in the dev set**, **unless** the authors show that the type of construction seen in that sentence is common in language F.
- up to 1 point for a general, clearly explained strategy that applies in 0 to 2 sentences **of the test set**.
- up to 1.5 points for a general, clearly explained strategy that applies in at least 3 sentences **in the test set**.
- up to 5 points for a general, clearly explained strategy that applies in **all sentences in the test set** and creates non-trivial improvement in the fluency and/or faithfulness of the system translation, relative to the baseline translation. It is the authors' responsibility to argue convincingly that there was non-trivial improvement. (Trivial improvement would be, for example, a single word being changed for a synonym.) An example of this type of strategy might be incorporating a language model to decide among candidate translations generated with other strategies.

Assign $\min(x, 10)$ (i.e., it is not possible to get more than 10 points).

Comparison with Google Translate (max. 4 points)

For each of the 5 test sentences:

- 0 points for lack of comment;
- up to 1.25 points for adequate comment on all the differences between the outputs (the students' system's and Google Translate's), not only pointing them out but indicating which is the better translation, and why.

Error analysis (max. 10 points)

- 0 points for general discussion of challenges without pointing to at least 2 specific errors;
- Up to 4 points for discussion of specific examples that does not trace at least 2 errors back to the design of the system;
- Up to 8 points for discussion of specific examples that traces at least 2 errors back to the design of the system

- Up to 10 points for discussion of specific examples that traces at least 2 errors back to the design of the system, and a feasible, insightful proposal of how to avoid those errors.

General quality of writing (max. 0 points)

This is not a writing class, so try to focus on the content; however, you may apply -4 points for writing that is **very difficult** to understand.

You've written 0 words

▶ [Preview next submission \(/cs124-](#)

[002/human_grading/view/courses/971938/assessments/8/peerGradingSets/8/peerGradings/1\)](#)

You are urged to preview the work of all 3 of your peers before you start submitting your required evaluations.

Remember to click the "Submit Evaluation" button after *each* time that you fill out an evaluation.

3 remaining of 3 required evaluations

[Save draft](#)

[Submit evaluation](#)