# Project Report

## Artificial Intelligence, Project-3

Rishi Josan (108996773) Ankit Dave (109364245) Abhiroop Dabral (108200755)

Our submission includes report and python files for all the questions.

1. **Q1:** For finding the missing letter, we have calculated the Probability by substituting the missing letter with all the possibilities from a-z and then we return the one which gives the maximum probability.
   To increase the efficiency, we are not calculating the part of the equation which will be constant for the all the possibilities. This increases the efficiency. Our algorithm runs in O(n).
   Output: Accuracy: 0.311102
   Time used: 0.269000 sec

2. **Q2:** For finding the missing letter, we have calculated the Probability by substituting the missing letter with all the possibilities from a-z. Now for all these values we substitute values from a-z for the second missing letter. Then we return the pair which gives the maximum probability.
   To increase the efficiency, we are not calculating the part of the equation which will be constant for the all the possibilities. This increases the efficiency. Our algorithm runs in O(n^2).
   Output: Accuracy: 0.360026
   Time used: 10.309000 sec

3. **Q3:** We first pre-compute CPT for words with one and two gaps. The first table stores probabilities for a gap of one letter, i.e. probabilities for a-a, a-b, a-c … z-z. We then use the previous table to compute another table that stores probabilities for words with two gaps, i.e. probabilities for a- - a, a- - b, a- -c, …. z—z .

   Pr(a-b) = Pr(a|a) * Pr(b|a) + Pr(b|a) * Pr(b|b) + Pr(c|a)*Pr(b|c) … + Pr(z|a)*Pr(b|z)

Pr(a- -b) = Pr(a-a)*Pr(b|a) + Pr(a-b)*Pr(b|b) + Pr(a-c)*Pr(b|c) --- Pr(a-z)*Pr(b|z)

For every word in the query we first add ` to the beginning and the end. We then count the number of "–" before and after the underscore. Two "-" correspond to using table 2, one "-" corresponds to table 1 and no "-" corresponds to the given CPT. We then multiply the two probabilities for a word for all characters and return the character which results in the highest probability.
Output: Accuracy: 0.148801
Time used: 0.332000 sec

4. **Q4:** For every character we compute the probability for all four words by following the process delineated in Q3 and multiply the four probabilities. We return the character that results in the highest combined probability

   Output: Accuracy: 0.405662
   Time used: 1.423000 sec

5. **Q5:** For finding the missing letter, we are finding the probability by considering the two before characters and two characters after the missing letter. We are substituting all the values from a-z and then we are returning the one with maximum value. Complexity of our algorithm is O(n).
   Output: Accuracy: 0.465159
   Time Used: 0.47300 sec

Output: Running all the questions together

```
$ python wordCross.py -q1 -q2 -q3 -q4 -q5
Question 1 accuracy: 0.311102
Q1 time used: 0.275000 secs.

Question 2 accuracy: 0.089656 (two correct letters)
Question 2 accuracy: 0.360026 (at least one correct letter)
Q2 time used: 10.199000 secs.

Question 3 accuracy: 0.148801
Q3 time used: 0.252000 secs.

Question 4 accuracy: 0.405662
Q4 time used: 1.383000 secs.
```

```
Question 5 accuracy: 0.465159
Q5 time used: 0.486000 secs.
```