Investigating the relationship between working memory and IQ

Matthew Byra
Kurt van Bendegem
COG260, University of Toronto

Abstract

Intuition suggests that given the task of immediately recalling a finite sequence of words a higher IQ would correlate with lower recall error rates, and faster recollection of terms. A statistical analysis was done on a data set of the performance of 141 individuals with a word recollection task and their corresponding IQs to provide evidence for the intuition. The 3 criteria for analysis were; modeling, fitting and finding the coefficient correlation of error rate and IQ, time taken to complete the task against IQ and the pace of response against IQ. The relationship with error rate and IQ gave provided a non-trivial correlation of -0.22 suggesting that IQ may decrease error in recollection.

Introduction

The dataset came from a study by Farrell, Simon, et al. named “Sequential Dependencies in Recall of Sequences: Filling in the Blanks.” Memory & Cognition. Its investigations hadn’t much to do with what we were interested in exploring. The dataset provided us with IQs of 141 Individuals, their response times to the task and there response time between word recollection. The subjects ran through over 200 trials individually summing to roughly 28000 total trials [1, pp.1555-1556]. Our hypothesis is that IQ provides faster and more accurate responses to recall tasks is consistent with the existing literature on IQ [2, p.268], [3, p.140], [4, p.167], [5]. However, there are few papers that explicitly compare IQ measures and working memory. Our alternative hypothesis is that general intelligence has no effect on the error rate and reaction of individuals performing immediate recollection tasks.

Methods

After interpreting the format of the data and performing some pre-processing, we split the analysis up into 3 separate measures:

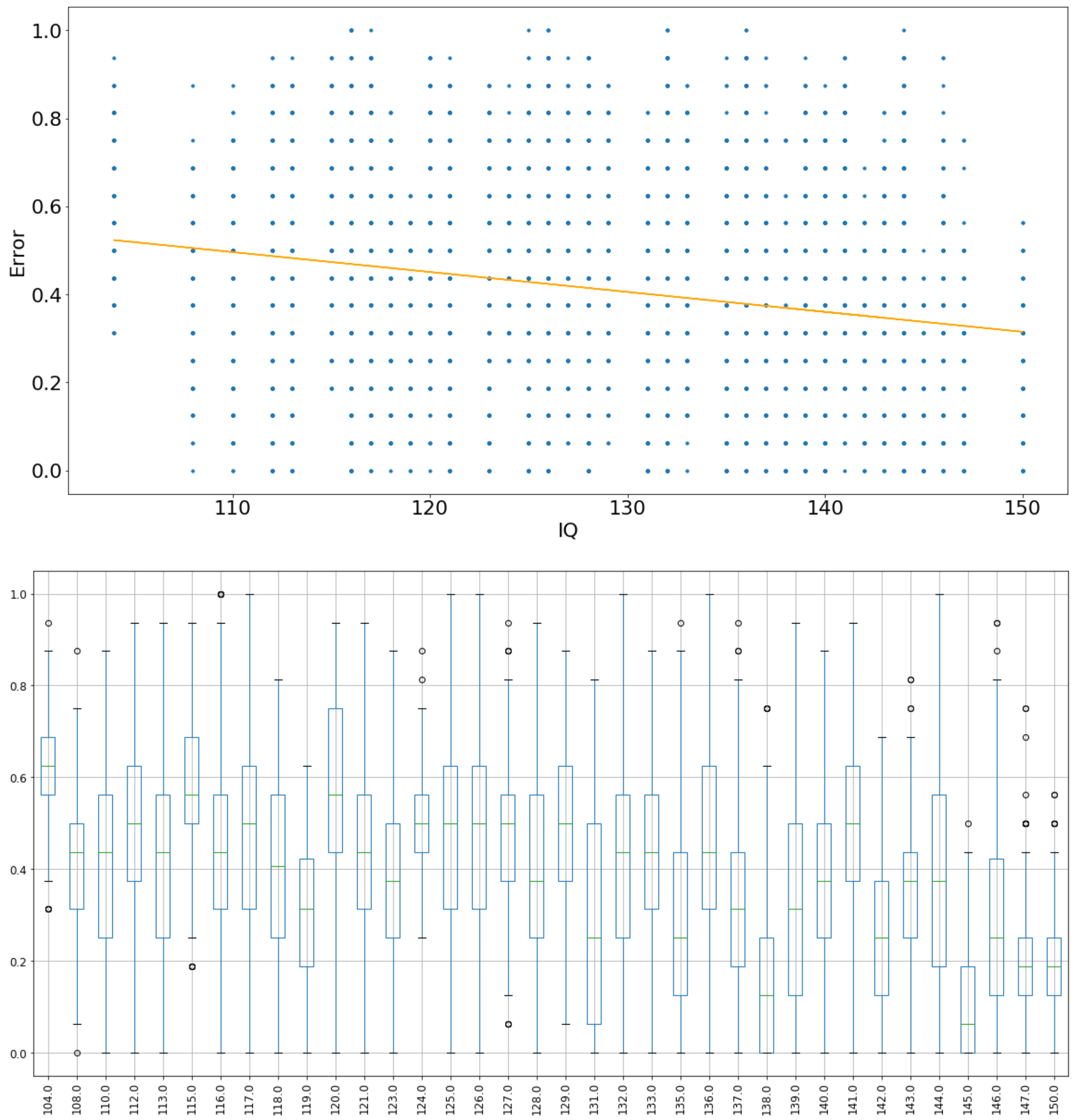
- Error rate of each trial (1 - percentage of correctly guessed words),
- Total time range for recalling all words in the trial,
- Average time between recalls in each trial

The data only records the given words and their ordering, as well as the times they were recalled, so these seemed like appropriate metrics to capture. For each of these 3, we generated a scatter plot of each with IQ as the dependent variable. We fit a line to each plot and also made note of the pearson correlation coefficient for each. In addition to the scatter plots, we also generated a series of boxplots, where each discrete IQ was given its own plot. This allowed us to analyze the mean and variance of each group, which provided clarity over the raw scatter plots. We also performed a permutation test on the correlations for the error measures, to determine how trivial the correlation coefficient we obtained might be.

Results

We will analyze the results in term of each of the three measures. First we have error rate:

`r = -0.22497`



Figures 1a and 1b: Error rate vs. IQ

We can easily observe a downward trend from the fitted line and the boxplots. The correlation value is quite low, but not trivial. This result is in support of our hypothesis, since IQ appears to be correlated with better performance here.

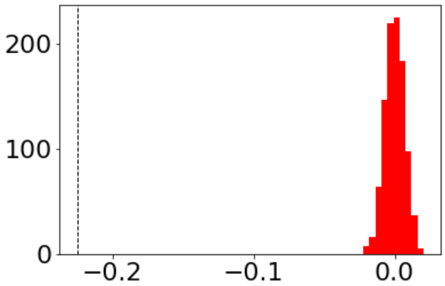


Figure 2: Permutation test for pearson correlation between Error and IQ

The result of the permutation test is shown above. The permuted trials center around zero, as expected. This test provided little insight, but illustrated the significance of the obtained correlation.

Now observe the plots for the next two performance measures:

r = 0.01804

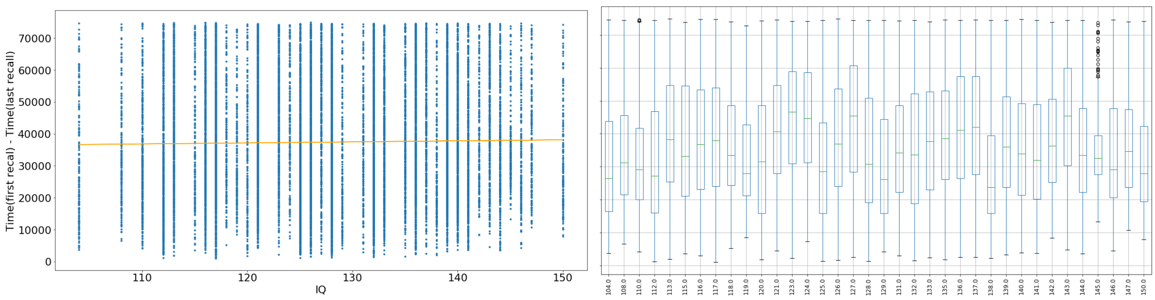


Figure 3a and 3b: Plotted data and Boxplot for total time taken for each trial

r = -0.07768

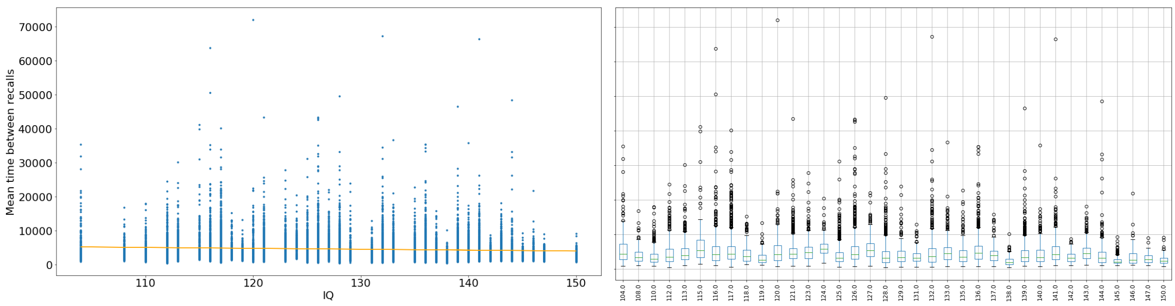


Figure 4a and 4b: Plotted data and Boxplot for mean word recall time

The two pairs of plots above indicate a very flat trend. There is an interesting 2 mode effect that we note in the boxplot of the second pair.

Conclusions

Overall, the results reveal fewer insights that we expected, given the huge size of this dataset. The notion of "performance" is difficult to reduce to one single metric, and we see different results for each of these. It is safe to say that there is a negative correlation between IQ and recollection error, within the range of IQs that this data offered. In the case of the time-related measures of performance, IQ has very little effect.

In light of these results, we reject our hypothesis, because overall performance was not improved across the board as IQ increased. Beyond this, we can make the conclusion that improved IQ does correlate with more accurate recollections of data. One possible initial extension of this study would be a broader range of IQ values. IQ tests are designed such that the values are normally distributed around a mean of 100, so all of the values in this data are above-average. This is a consequence of the study using university students as test subjects exclusively. Its possible that some lower bound on recall time is reached at an IQ range below 100, which our data would not illustrate. Another interesting path to explore in an extended study would be an investigation on the relation between the actual content of the words themselves and the IQ of the participant. Unfortunately Word data was data that went unused for the purposes of this study.

References

1. Farrell, Simon, et al. "Sequential Dependencies in Recall of Sequences: Filling in the Blanks." Memory & Cognition , vol. 41, no. 6, 2013, pp. 938–952., doi:10.3758/s13421-013-0310-0.
2. Roediger, Henry L. "Inhibiting Effects of Recall." Memory & Cognition , vol. 2, no. 2, 1974, pp. 261–269., doi:10.3758/bf03208993.
3. Baddeley, Alan D. "Working Memory". Clarendon Press, 1995.
4. Blankenship, Tashauna L., et al. "Working Memory and Recollection Contribute to Academic Achievement." Learning and Individual Differences , vol. 43, 2015, pp. 164–169., doi:10.1016/j.lindif.2015.08.020.
5. Hembree, Ray. "Experiments and Relational Studies in Problem Solving: A Meta-Analysis." Journal for Research in Mathematics Education , vol. 23, no. 3, 1992, p. 242., doi:10.2307/749120.

Appendix

Python Code

```
#!/usr/bin/env python
# coding: utf-8

# In[1]:


import string
import numpy as np
from scipy.stats import pearsonr
import matplotlib.pyplot as plt
import pandas as pd

presented_items_df = pd.read_csv('./healetal_data_csv/pres_items.csv')
recalled_items_df = pd.read_csv('./healetal_data_csv/rec_items.csv')
recalled_items_df = pd.read_csv('./healetal_data_csv/rec_items.csv')
recall_times_df = pd.read_csv('./healetal_data_csv/times.csv')

# build data frame for subject data
trial_to_subjects = pd.read_csv('./healetal_data_csv/subjects.csv')
num_trials = len(trial_to_subjects)
subjects = pd.DataFrame()
subjects_array = pd.unique(trial_to_subjects['subject'])
iqs =  pd.read_csv('./healetal_data_csv/fsiq.csv')

subjects["subject"] = subjects_array
subjects["iq"] = iqs["fsiq"]
subjects_to_iq = {}

for index, row in subjects.iterrows():
    subjects_to_iq[int(row['subject'])] = row['iq']

presented_items_df["subject"] = trial_to_subjects
recalled_items_df["subject"] = trial_to_subjects


# In[2]:


trial_errors = np.zeros((num_trials))
trial_iqs = np.zeros((num_trials))
trial_time_ranges = np.zeros((num_trials))
trial_mean_intervals = np.zeros((num_trials))

#for each trial, find all of the error rates.
for trial in range(num_trials):
    cur_errors = 0
```

```python
    subject_id = (presented_items_df.iloc[[trial]].values)[0][-1]
    iq = subjects_to_iq[subject_id]

    # calculate error percentage
    presented_words = (presented_items_df.iloc[[trial]].values)[0][:-1]
    recalled_words = (recalled_items_df.iloc[[trial]].values)[0][:-1]

    presented_set = set(presented_words)
    recalled_set = set(recalled_words)

    unrecalled_words = presented_set.difference(recalled_set)

    trial_errors[trial] = len(unrecalled_words)/16
    trial_iqs[trial] = iq

    # calculate time ranges
    times = recall_times_df.iloc[[trial]].values[0]
    times = np.trim_zeros(times)
    if len(times) >= 2:
        time_range = times[-1]
    else:
        time_range = None

    trial_time_ranges[trial] = time_range

    # calculate the average time between recalls
    if len(times) >= 2:
        mean_time_interval = np.mean(np.diff(times))
    else:
        mean_time_interval = None

    trial_mean_intervals[trial] = mean_time_interval


# In[4]:


# plot each performance result against IQ values.
iqs_clean = trial_iqs[~np.isnan(trial_iqs)]
errors_clean = trial_errors[~np.isnan(trial_iqs)]

range_iqs = trial_iqs[~np.isnan(trial_iqs) & ~np.isnan(trial_time_ranges)]
ranges_clean = trial_time_ranges[~np.isnan(trial_iqs) &
~np.isnan(trial_time_ranges)]

interval_iqs = trial_iqs[~np.isnan(trial_iqs) & ~np.isnan(trial_mean_intervals)]
mean_intervals_clean = trial_mean_intervals[~np.isnan(trial_iqs) &
~np.isnan(trial_mean_intervals)]

cleaned_length = np.shape(errors_clean)[0]
```

```python
def plot_data(iqs, data, xlabel, ylabel):
    plt.rcParams.update({'font.size': 22})
    correlation = pearsonr(iqs, data)
    print("Correlation: " + str(correlation[0]))

    fit = np.polyfit(iqs, data, 1)

    plt.figure(figsize=(20,10))

    plt.xlabel(xlabel)
    plt.ylabel(ylabel)
    plt.scatter(iqs, data, s=10)
    plt.plot(iqs, iqs * fit[0] + fit[1], color='orange')
    plt.show()

# error plot
plot_data(iqs_clean, errors_clean, "IQ", "Error")

# time ranges plot
plot_data(range_iqs, ranges_clean, "IQ", "Time(first recal) - Time(last
recall)")

# mean time interval between recall plot
plot_data(interval_iqs, mean_intervals_clean, "IQ", "Mean time between recalls")


# In[49]:


# Generate a boxplot of the error data.
def generate_iq_boxplot(trial_data, iqs):
    boxplot_df = pd.DataFrame()
    iq_to_trial_data = {}

    for i in range(len(trial_data)):
        error = trial_data[i]
        iq = iqs[i]
        if not np.isnan(iq):
            if iq in iq_to_trial_data.keys():
                iq_to_trial_data[iq].append(error)
            else:
                iq_to_trial_data[iq] = [error]

    max_length = 0
    keys = list(iq_to_trial_data.keys())
    for i in range(len(keys)):
        if len(iq_to_trial_data[keys[i]]) > max_length:
            max_length = len(iq_to_trial_data[keys[i]])
```

```
    keys = list(iq_to_trial_data.keys())
    keys.sort()

    for iq in keys:
        boxplot_df[iq] = iq_to_trial_data[iq] + ([None] * (max_length - 
len(iq_to_trial_data[iq])))

    plt.rcParams.update({'font.size': 12})
    plt.figure(figsize=(20,10))
    boxplot_df.boxplot(rot=90)

generate_iq_boxplot(trial_errors, trial_iqs)
generate_iq_boxplot(trial_time_ranges, trial_iqs)
generate_iq_boxplot(trial_mean_intervals, trial_iqs)


# In[5]:


# permutation test.
actual_correlation = pearsonr(iqs_clean, errors_clean)[0]
print("Actual correlation: " + str(actual_correlation))

num_permutations = 1000

permuted_correlations = np.zeros(num_permutations)

for i in range(num_permutations):
    shuffled_iqs = np.random.permutation(iqs_clean)
    cor = pearsonr(shuffled_iqs, errors_clean)[0]
    permuted_correlations[i] = cor

plt.hist(permuted_correlations, color='red')
plt.axvline(actual_correlation, color='k', linestyle='dashed', linewidth=1, 
label="actual correlation")
plt.show()
```