Project Report - Language, Computation, and Cognition (096222)

Roni Fridman – 205517097
Tameer Milhem – 322729443

GitHub Link to Repository: Link

Project 1 – Surprisal and Reading Times

Abstract

This study investigates the relationship between surprisal and reading times, utilizing both structured and semi-structured tasks to analyze different computational models, including 5-gram models, RNNs, and Generalized Additive Models (GAMs). Surprisal, a measure of unpredictability in language processing, is tested against reading times to determine its effectiveness in capturing cognitive load during reading. Through experiments on the Penn Treebank and Wikitext-2 datasets, we explore the predictive power and generalization ability of these models. Results indicate that while both n-gram and neural models provide insight into surprisal's effects, their performance varies significantly across contexts, with evidence suggesting that the relationship between surprisal and reading time is predominantly linear but influenced by additional linguistic factors such as word frequency and length. This work contributes to our understanding of how different models perform in predicting reading times and highlights the importance of using multiple models and control variables to achieve a comprehensive analysis.

Introduction

The study of surprisal in computational linguistics provides insights into the cognitive processes involved in reading. Surprisal quantifies the predictability of words within a sentence and is derived from probabilistic models of language. This report addresses key questions in understanding how surprisal affects reading times and compares different model approaches, including NGRAM and RNN models.

Previous research has shown that surprisal can predict reading times, reflecting cognitive load during reading (Hale, 2001; Levy, 2008). Our project aims to explore this relationship using both structured and semi-structured tasks, and to investigate the utility of combining multiple models in an open-ended task. This study is novel in its comparison of different model types and its application of a Multiple Experts approach, providing new insights into how well these models generalize across different datasets.

Data

The project utilized the following datasets:

- **Penn Treebank**: A well-established dataset in natural language processing, which provided reading time data.
- **Wikitext-2**: A larger dataset used to train and evaluate models for a richer set of language contexts.

Key statistics:

- **Penn Treebank**: Contains reading time data aligned with surprisal values.
- **Wikitext-2**: Large-scale dataset for model training, consisting of Wikipedia pages with reading times.

Experiments and Results

<u>Structured Task</u>

We trained an RNN and a 5-gram model on the Penn Treebank data. After aligning the surprisal values with reading times, we compared the performance of the two models.
We plotted the both models and used a Pearson correlation coefficient to measure correlation between the surprisal values each model has produced and the reading times. The 5-gram model had a coefficient of R=0.0043 while the RNN reached R=0.0078. While not a large difference, we could see in the plotted chart that the 5-gram model tends to underestimate the delay of reading time, especially in the higher surprisal values, while the RNN encapsulate those slightly better.

Next, we wanted to understand where these models differ in general. We did that using first plotting a density chart between both surprisal values for the same word-code pairs, as seen in Figure 1.

We can see that the models tend to agree for their surprisal values, especially in the low-mid values (5-7.5). Their main difference arises from the scale – it seems as the 5-gram model didn't use a surprisal value greater then 20 and we see that the RNN could indeed reach those values, although as to be expected, rarely.

Next we tried to find examples of specific words that the models disagree on to get a clearer picture. We wanted to showcase one example of each disagreement – the first is when the 5-gram model assigned a higher surprisal value then the RNN model. One such word was "feet" from text number 6: "There's tower steeple on church million feet high". We can see that the 5-gram model gave a high surprisal score compared to the RNN model since use of the word can be both a unit of measurement and a body part, and only texts written in the USA will use feet as a unit of measurement. An example for when the RNN has assigned a higher value was more common, and we chose the word "stream" in text 4: "kept up steady stream pseudo-scientific mumbo-jumbo." It seems the word stream, just like the word now, is very frequent in it's original use, and the RNN model find the disambiguate words that are very frequent. For this reason, the RNN model also fails with a lot of stop words such as He, His, She, is, the and the like. In general, it seems that when there is a disagreement the RNN is usually the one to score higher in surprisal and we suspect might be relying too heavily on the frequency of the word.

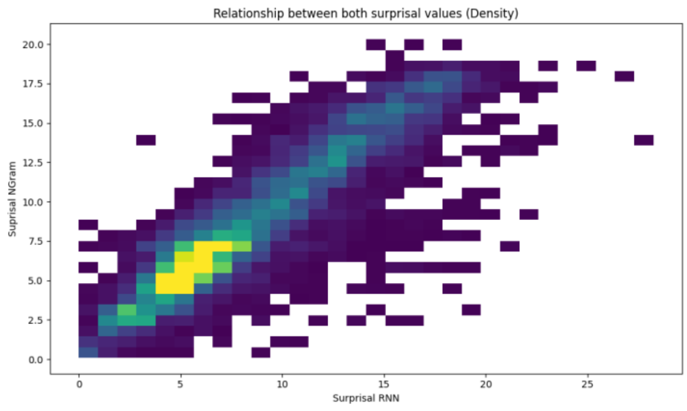

Figure 1: Density between surprisal values produced by RNN and 5-gram models

Next we wanted to see the spill-over effect in action. We plotted the probability the model used (calculated as $p(x) = 2^{-surprisal(x)}$) vs the reading time. The results showed something interesting as seen in Figure 2. We decided to show only RNN since the 5-gram chart is very similar.

We notice that in general the spill-over effect holds - the next word's RT is larger then the current's. What is surprising is that at the extreme for very rare words with low probability it shifts. It does makes sense - word that are so unexpected can be totally unknown for the readers, or so special readers try to understand how to pronounce them or why they were even used at the same moment they read it.

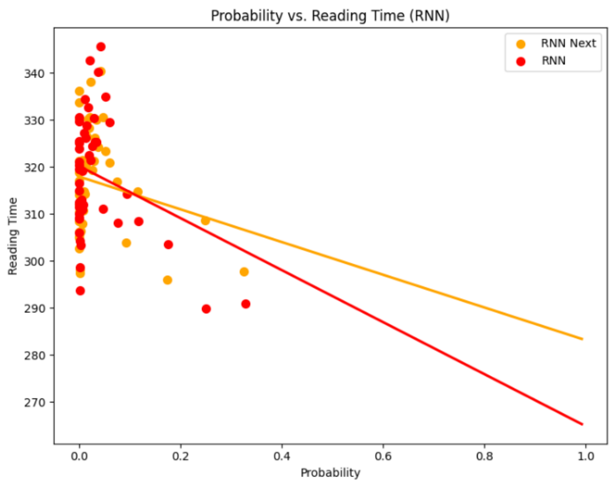

Figure 2: The spill-over effect holds until a low enough probability

<u>Semi-Structured Part 1: Fitting and plotting the RT surprisal curve using a General Additive Model (GAM)</u>

We wanted to evaluate the relationship between surprisal values and reading times. In the article of (1) the claimed for a linear connection. We wanted to test the claim.
For that purpose, we created a function that configures A GAM model that is trying to fit the best linear model using the variables it was presented with (mainly, the surprisal values). We also used control parameters (word log-frequency and word length) to understand their added value and effects. Since the dataset is rather small, we used the "wordfreq" library for the words frequency, and because of this we needed to process the words differently (we just used lower case and removed all symbols that are not letters) since the frequency will not mean anything for tokens such as <unk> which appear in the text for every word with an apostrophe inside of it. We then plot the results of the real data vs the prediction of the GAM model to see how well it

preforms.
When plotting the real data, we decided to use a transformation - we divided by 15 (after trial and error) and subtracted the median of the reading time. It's interesting to see that the data fits very well afterwards regardless of the model or control variables chosen. Since in general this is close to a linear transformation (we could've used mean instead of median but it was worse for some reason) and the GAM model will fit the model in a linear manner with different denominators, It's the shape of the model that count as long as the transformation is linear. Results were very similar between RNN and 5-gram model, so we'll only showcase the RNN model here.

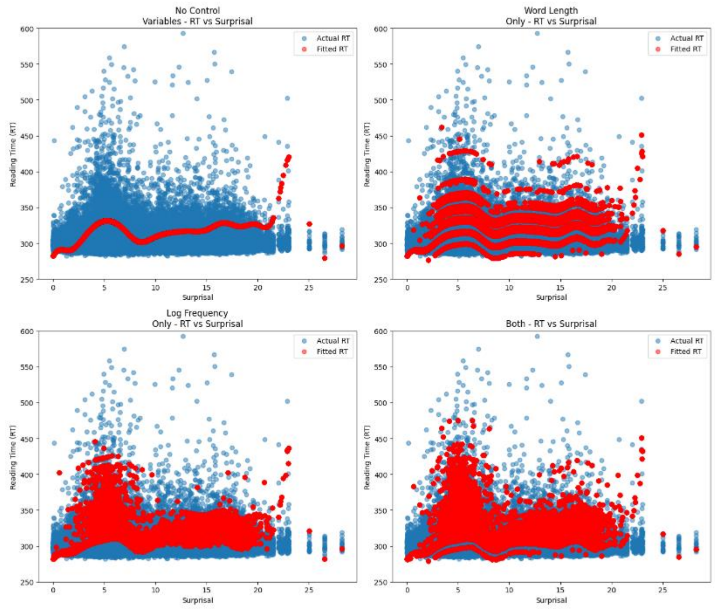

Figure 3: Real data in blue, model prediction in red. Top left – No control variables, Top right – word length as control, bottom left – log-frequency as control, bottom right – both variables as control.

We'll start with the model with no control variables. We can see the line does encapsulate the general form of the actual data - but it's missing a lot of variability. After adding the word length variable, we can see lines are forming - this makes sense since it's a discrete variable and there's a connection between word length and reading time as presented in (2). We can see it indeed captures a lot of depth the naked model with no control variables couldn't. If we use only word frequency (bottom right) we can see it's added value compare to the word length - word length is giving more information about the amplitude of the reading time - how extreme the RTs will get, while the frequency focuses on the accuracy by being continues and characterizing every word instead of its group (i.e., "good" belongs to the group of four letter words).

Finally, in the model containing both control variables we can see exactly that - the model is much denser compared to the "Word Length Only" model by utilizing the frequencies, and much higher and extreme (when needed) compared to the "Log Frequency Only" model - utilizing the word length column.

Regarding the weigh of the control variables, we already notice the weight the GAM model has shown us for every control variable - every control variable is giving a lot of information compared to a "naked" model with only surprisal values. Although each variable contributes a lot, we can see the contribution of moving from one control variable to both of them improves the predictions only by a bit.

This means that the variables are not independent, and that makes sense, since people tend to use shorter words more frequently. That means the information we gain from both variables is overlapping quite a bit. Nevertheless - we could still capture through the graphs what each one contributes uniquely.

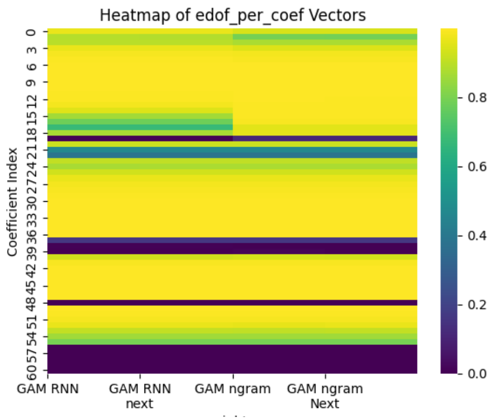

Figure 4: Splines coefficients of the GAM models

We printed all the statistical values of the GAM models we computed. First, we can see that the p-values are very small - meaning that the variance that's explained by the variables we chose is significant. That being said, we can see that it fails to explain a lot of the variance. we saw since the R-squared in all of the models is very small, with the highest value in the GAM RNN Next model (spill-over) with a explain deviance of 0.0143. We saw this in the structured part as well, and the way the data was processed to train the model makes it very noisy to say the least. In figure 4 we can see the coefficients of each variable. The GAM model uses splines – smooth functions that try to encapsulate the linearity between the variables and target. We used the default 20 splines for each variable and figure 4 present the model with both control variable for both model architectures (RNN and 5-gram) for both the regular RTs and for the spill-over effect. The last spline is for the "error" variable of the model. Recall the order is: 0-19 for "surprisal", 20-39 for "log-frequency", 40-59 for "word length" and 60 for "error". We'll claim for evidence about a linear connection between suprisal and reading time values:

- Argument for Linearity: The fact that most edof_per_coef values are near 1 suggests that the relationship between surprisal and RT is being modelled almost linearly in most splines. Coupled with a low explained deviance, it implies that additional non-linear terms may not be substantially improving the model, supporting the argument for a linear relationship.
- Argument against Non-linearity: If the relationship were highly non-linear, we would expect higher edof_per_coef values, usually higher then 1, and potentially better fit statistics, which both aren't strongly evident in this case.

In conclusion, although the data we had is noisy and there's more to the RT that only surprisal values, word length and word frequency, the do play a major part in the deviation of the reading time and we support the belief that their connection is mostly linear (can be modelled linearly).

<u>Semi-Structured Part 2: Using Wikitext-2 as an Alternative dataset</u>

The preprocessing routine is designed to clean and standardize the text, preparing it for further analysis. First, titles surrounded by sequences of equal signs (=) are reformatted by removing the = signs, converting the titles into standard text. Next, the text is filtered to retain only specific characters, including letters, numbers, brackets, commas, semicolons, periods, apostrophes, and hyphens, while removing any extraneous characters to maintain consistency.

Possessive forms like "word 's" are then replaced with the placeholder to normalize these phrases throughout the text. Additionally, any empty lines are removed, ensuring that the text is compact and continuous. Finally, symbols such as @-@ and @,@ are replaced with the correct characters (- and , respectively) to ensure proper representation of hyphens and commas.

We aimed to train a 5-gram model, similarly to out previous tasks, but we found that almost all words got a surprisal value of 33.219. This is likely due to the domain mismatch between the training and testing data, where the 5-gram model was trained on Wikitext2, which has a different vocabulary and sentence structure compared to Penn Treebank. This leads to many out of vocabulary tokens and unseen 5-grams when applying the model to the Penn Treebank data. As a result, the model assigns low probabilities to these unfamiliar sequences, resulting in high surprisal values. also, the sparsity inherent in a 5-gram model make this issue worse as it requires specific sequences that are rare or absent in the new data.

Since the 5-gram model is not very informative here, we decided to switch to a bigram model.

The results show that the bigram and RNN model perform poorly on the Penn Treebank dataset, with very low correlations between surprisal and reading time ($r \approx 0.004$-0.008), this suggests that both models particularly the 5-gram model, struggle with generalization due to domain mismatch and n-gram sparsity. In contrast, on the Wikitext-2 dataset where both the bigram and RNN models were trained, the correlations improve slightly, especially for the bigram model ($r \approx 0.093$), indicating a better (though still weak), alignment with reading times. The bigram model stronger performance may reflect its robustness in handling simpler more frequent patterns, whereas the RNN, while more complex, still shows weaker generalization likely due to its reliance on more specific contextual information.

Open-Ended Task

For our open ended task, we wanted to simulate different models as experts for the suprisal of a word. Meaning, if different architecture can detect different patterns in a word's surprisal value, and if so, can we use simple yet effective methods to create a better more informed surprisal values.
This idea is similar to the weak-learners approach, where we have several different weak learners – meaning they are simple and have low complexity which mean low computational cost, but every model on it's own is not providing a very good answer to a task – And yet, when combined and aggregated, they can potentially turn into a very reliable model.

With regards to our task we first needed to choose metrics of evaluation in order to define when are models preforming better. Since we suspect the connection between the surprisal value and the reading time are linear, based on other research and what we found in task 1 of the semi-structured part and the paper by Levy and Smith (1), our evaluation metrics were:

1. Correlation (R-value) between the Surprisal and RT. A higher value means higher linear correlation, which is closer to the truth.

2. MSE of Predictions – Based on using the surprisal value as a feature in the GAM model, we want the new surprisal value to be able to predict the reading time more accurately. We chose MSE as an error rate since it's easy to explain and was used in the GAM model as well.

First, we train an LSTM model both on the Wikitext and the Treebank datasets, in addition to the two bigram and RNN models trained on each dataset, resulting in 6 models. While we're aware they are similar architectures, the results we saw in the semi-structured task 2 showed a significant change between them.

We then merged and aligned the data to receive the full table of word-code pairs, reading time, and the six surprisal values calculated by each model. A big problem we observed was the same we saw when training the 5-gram on Wikitext – we had a lot of values that were not seen by the model and received the default value of 33.219281. We thought about preprocessing the values, but decided this is one of the reasons the models are "weak learners" – one of their innate flaw, and it should be part of the process.
That being said, we did chose to remove them from certain calculations when it was deemed necessary.
We noticed that this decision came with a cost – it seems models tend to underestimate the surprisal value in general, meaning they assign a probability that although small, should be smaller. This means that the bigram model, that had a lot of very high surprisal values caused by out-of-vocabulary terms was seen as highly correlated with the reading times. When we used to preprocess and replace the default values with the mean of all surprisal values, the correlation changed – most significantly in the bigram Wikitext model, the value dropped from 0.0947 to 0.0664. On the other hand, the LSTM Wikitext model R value raised from 0.0778 to 0.0862, which was the highest value.

We wanted to start by looking at the behaviour of simple aggregation functions on the values the models have produced. We made sure the aggregation is only on cells that do not have the default value, since it with only 6 models, such a large value will make the calculation too noisy.
We can see the results in figure 5 – recall that we decided to remove the default values from calculations, so that means the highest single model was 0.0862 for the LSTM Wikitext model, and the only aggregate function that produced an R value > 0.08 is Max – which steel-mans our belief that the model are underestimating the surprisal values.

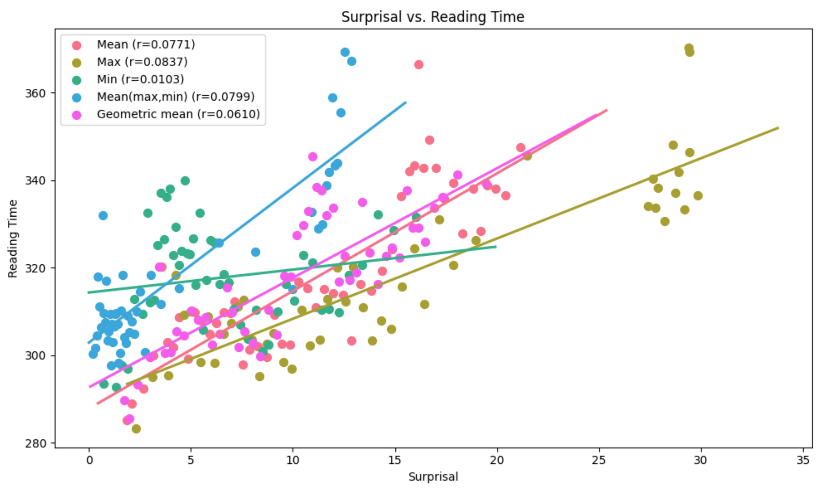

Figure 5: Surprisals derived from simple aggregation functions

Next we wanted to train linear models for two reasons – the first is we wanted to see how would the simple aggregations would fare in predicting the reading time values (our second metric), compare to the original model (compromised of all six surprisal values) based on MSE. The second reason was for deciding the weights of the model.
Linear model have high explainability, which means we can see which surprisal values are more effective and informative, and thus, Create a weighted mean using the feature importance every linear model attirubutes the every surprisal value. We can see the results in Figure 6 for the three models we chose: Linear Regression, Decision Tree and Random Forest.

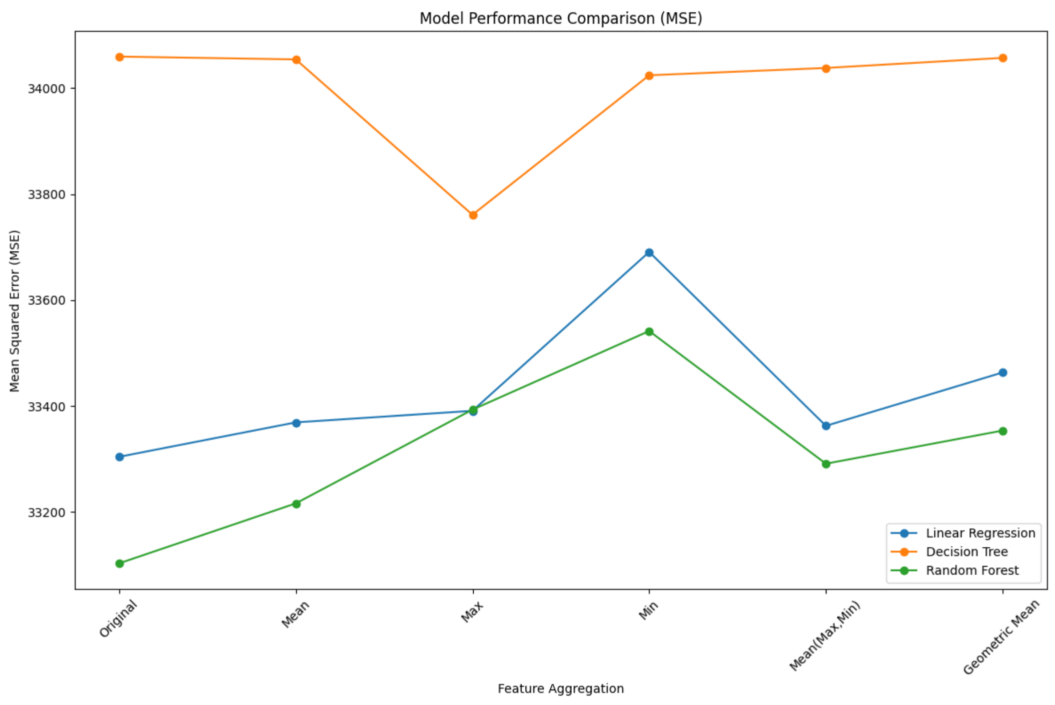

Figure 6: MSE for every Linear model trained on different aggregations of surprisal values.

We can see here that the decision tree seemed to be a lot less informative. Since it is very similar to Random Forest we will focus on the other two linear models. We cam see that compared with one another, it seems the simple functions, although not bad, cannot reach the original model's MSE values, which is logical since it contains more information than the others.
Still, we can see that these models preform much better than the GAM model which was always above 40,000K MSE. The main reason for it must be related to the fact the most of these models filter out the default value in their aggregation.
Now we can see the value of importance each linear model has assigned to each surprisal value in Figure 7. what each linear model and try to use these as weights for the new weighted means surprisal values:

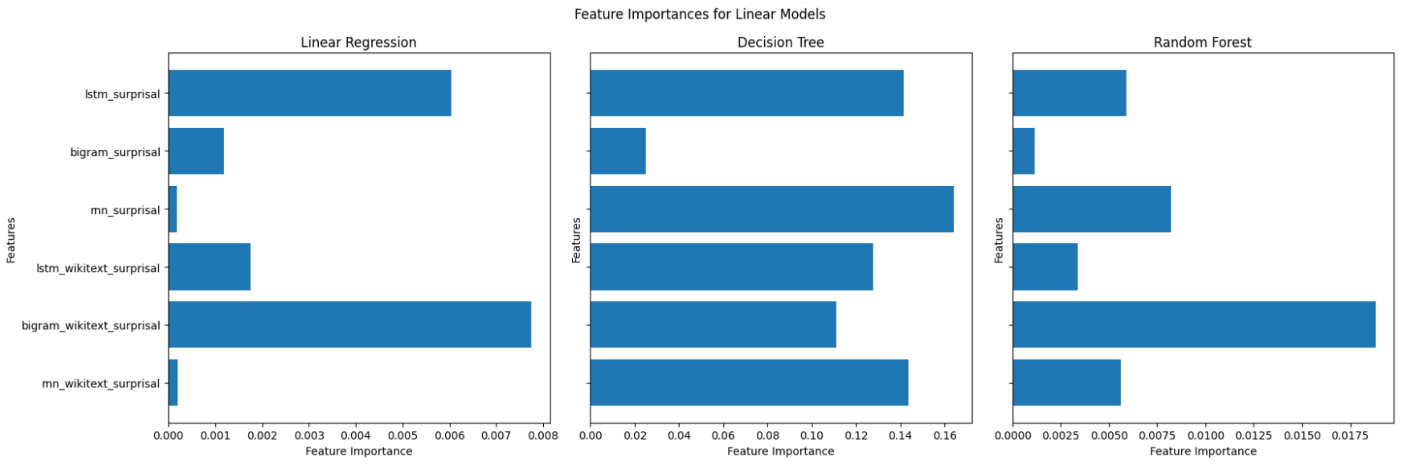

Figure 7: Feature importance by LR, DT and RF linear models for each surprisal value

We'll note that in these models, since they are more sophisticated compare to the simple aggregation functions we decided not to filter out the default values. This resulted in a very high importance by the bigram Wikitext model, which as we already discussed contains a lot of these values. We can see that the LSTM model was deemed as important by both LR and RF models, compare to the RNN. This makes sense since we expected it to behave better, since the LSTM is a type of RNN but is able to handle longer inputs better.
Now we can create the final surprisal values as the average mean of each model and compare the results.

We can see in Figure 8.b the linearity of the models. We can clearly see the R values are much higher compare to previous model (both when we filtered the default values and when we didn't) which means that they indeed result in a more linear relationship between the two variables.
If we look at Figure 8.a, and Figure 8.c, we can see the MSE of each weighted mean is better then each single surprisal value on its own, and even better then the MSE of the values from the simple aggregations.

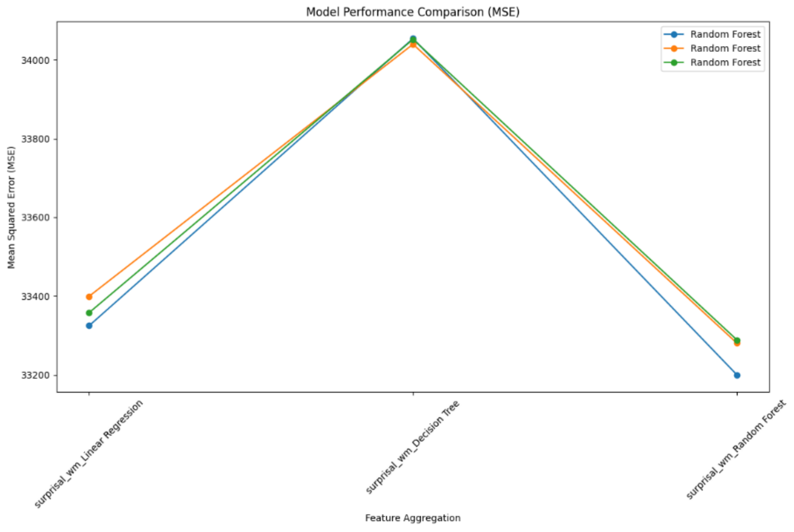

Figure 8.a: Feature importance by LR, DT and RF linear models for each surprisal value

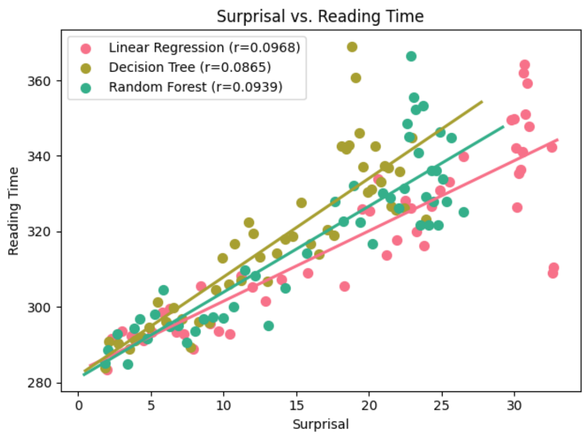

Figure 8.b: Feature importance by LR, DT and RF linear models for each surprisal value

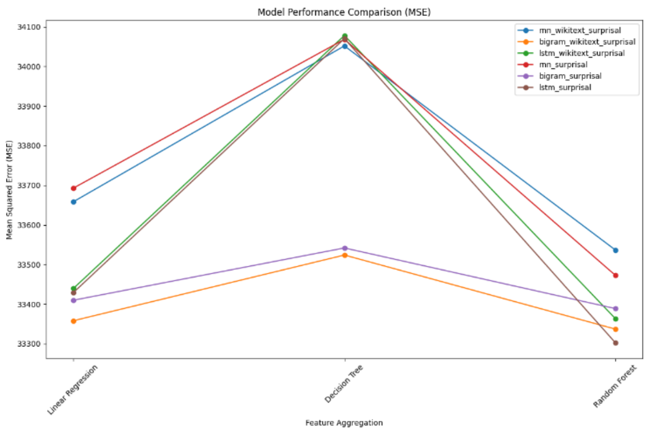

Figure 8.c: Feature importance by LR, DT and RF linear models for each surprisal value

Each model, as a feature, was able to reach on it's own a minimal MSE of no less then 33,300, while the weighted mean of the RF model was able to reach lower then 33,200 MSE. We can also see that no model was able to reach the 33,100 that the original model (the one compromised from all six values) was able to reach.
This makes a positive argument for our case – which is that one model type is not enough, and some models are inherently able to find patterns or even blind spots other models, even deep learning models, are missing compare to more simple models.
If utilized right, this could result in better results and lower computation cost for RT prediction tasks, and since these simple models have high explainability, can help and shed light on the cognitive nature behind reading time w.r.t the context red.

Discussion and Conclusions

Our findings suggest that surprisal, as predicted by both traditional n-gram and more advanced RNN models, shows some correlation with reading times, indicating its relevance as a measure of cognitive load in reading comprehension. However, the differences in model performance reveal critical insights. The RNN model, though slightly better at handling high-surprisal words, often relies too heavily on word frequency, causing mispredictions, especially for highly frequent or contextually ambiguous words. The 5-gram model, on the other hand, suffers from issues of sparsity and domain mismatch, particularly evident when trained on Wikitext-2 and tested on the Penn Treebank dataset. This results in less robust predictions and an overestimation of surprisal for unseen words or sequences.

The semi-structured task using Generalized Additive Models (GAMs) provided further evidence supporting the linear relationship between surprisal and reading times. While the GAM models with control variables such as word length and log-frequency added depth to our analysis, they also highlighted the interdependence of these variables, implying that while each contributes uniquely, their combined effect is not entirely independent. This is particularly evident in the overlapping variance explained by these variables. The low R-squared values across all models suggest that while surprisal, word length, and frequency are significant factors, they do not fully account for the complexity of reading time variation. This points to the need for more nuanced models that consider additional cognitive and contextual factors.

Our exploration using the Wikitext-2 dataset further demonstrated the limitations of simpler models like the bigram, which showed slightly better performance on familiar data but struggled with generalization when faced with domain shifts. This reinforces the idea that while more sophisticated models such as RNNs can capture complex dependencies, their reliance on specific contexts can hinder their effectiveness in broader applications.

In conclusion, this study underscores the utility of surprisal as a predictive metric for reading times but also reveals the limitations of existing computational models. A multiple-experts approach, combining the strengths of various models, may provide a more comprehensive understanding of reading behaviours. Future research should explore hybrid models and consider integrating more diverse linguistic and cognitive features to enhance prediction accuracy. Additionally, more robust cross-domain training and evaluation frameworks are needed to improve the generalizability of these models across different linguistic datasets.

Bibliography

1. *"The effect of word predictability on reading time is logarithmic".* **Levy and Smith.** 2013.

2. *A theory of reading: From eye fixations to comprehension.* **Just, M. A., & Carpenter, P. A.** 1980, Psychological Review.

3. *A probabilistic early parser as a psycholinguistic model.* **Hale, .** 2001, Proceedings of the 6th Conference on Computational Natural Language Learning.

4. *Expectations and inference in reading comprehension.* **Levy, .** 2008, Cognitive Science.