Problem Chosen	2023 MCM/ICM Summary Sheet	Team Control Number
C		2309900

Unleashing the Power of Word Embeddings: From Word Vectors to WORDLE

Summary

A simple word game, **Wordle**, has become the newest social media and pop culture phenomenon. In this paper, We established several models, including Model I: Temporal 2D-Variation Model, Model II: Convolutional Transformer with Word2Vec, and Model III: Difficulty Classification Based on Model II and Statistics.

For Model I, in order to accurately predict the number of players, we established a **Temporal 2D-Variation Model**. We extended the analysis of temporal variations into 2D space by transforming the 1D time series into a set of 2D tensors based on multiple periods, allowing our model to overcome the limitations of 1D time series in representation capability. We used RMSE and MAPE as measurement indicators and found that our model **outperformed ARIMA and LSTM**. We predicted that there will be 20.19956K-20.98455K players on March 1. We also found that the curve fitted follows a gamma distribution, and that word attributes do not affect the percentage of players in difficulty mode.

For Model II, we established a predictive model: **Convolutional Transformer with Word2Vec**, which can transform input words into embeddings and extract each feature of the word as accurately as possible. We used PCA to fuse word features into a matrix and input the matrix into the ConvTransformer to obtain accurate results. **Quantile regression** was used to verify this model and we found that it is 95% reliable. We also calculated the percentage of the word ”EERIE” and obtained the results.

For Model III, which is a supplement to Model II, we added a classifier to realize word difficulty classification. We used **SGDM** to obtain the attribute weight for each word. After calculating the comprehensive score of each word, we classified them into four categories: Beginner Level, Challenger Level, Master Level, and Legendary Level based on scores. After training for 50 epochs, we used **F1-score** to evaluate the model and achieved an accuracy rate of 94%. The model determined the difficulty of the word ”EERIE” is Legendary Level.

Finally, we listed some interesting features of the dataset. Our model has strong **adaptability**. It is not only applicable to the analysis of Wordle but also to predict the majority of time series. It can handle input in the form of words, phrases, or articles, and provide high-precision classification and prediction results.

Keywords : Temporal 2D-Variation; ConvTransformer; SGDM

Contents

1 Introduction

1.1 Problem Background

The trivia game **Wordle**[1], which is placed in the New York Times every day on time, requires each player to guess a five-letter word in the final question by taking up to six guesses to get clues about the letters of the word and their positions.

We collected some data from Twitter[2], through which we observed that the number of users and their scores trended accordingly as time changed and the word characteristics of the answers varied daily. The phenomenon above piqued our interest. So we decided to use this data to develop several specific models to study and calculate the relationship between their trends deeply.

1.2 Restatement of the Problem

Considering the background of the question and its limitation, we decide to focus on the following questions:

- Develop a model that explains the daily variation in reported results and use it to create predictions. Investigate whether any attributes of the word affect the percentage of scores reported in hard mode.
- Create a model that predicts the distribution of reported results for a given solution word. Give the uncertainty and confidence of the model.
- Summarize a model that classifies solution words by difficulty and identifies the attributes associated with each classification. Give the classification results and accuracy of the model.
- Analyze any other interesting features of this data set.
- Write a letter to the Puzzle Editor of the New York Times to summarize our model and results.

1.3 Our Works

This problem requires a prediction and classification model to analyze Wordle's data. Our work mainly includes the following:

- Based on words, time, and the number of reported results, we established a Temporal 2D-Variation Moele for time series prediction of the number of reported results, which can tackle the intricate temporal variables with the help of the covariate "word" .
- Quantify the attributes of words and test the correlation of the attributes.
- We established Convolutional Transformer with Word2Vec for regression and classification. The model can well encode and extract the characters of the input words and achieve high precision in its prediction results.
- Through analysis, we present evidence that our model provides the best results.

In order to avoid a complicated description and intuitively reflect our work process, the flow chart is shown in Figure 1:

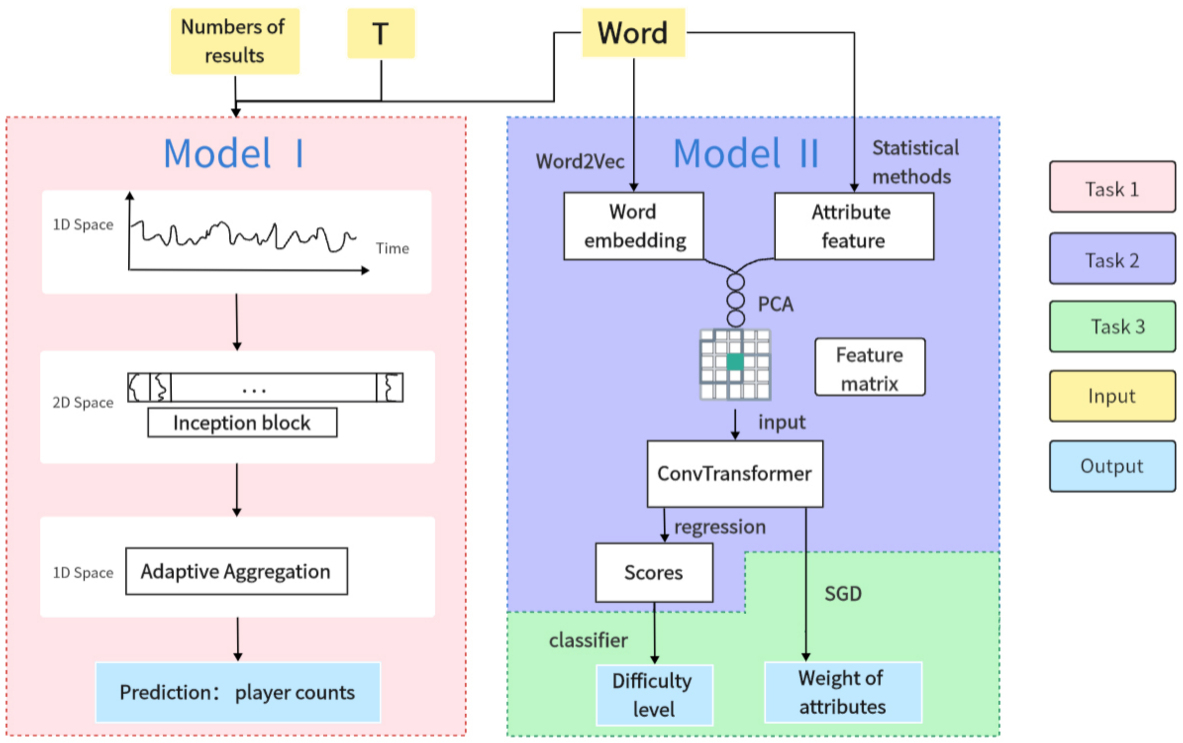


Figure 1: Flow Chart of Our Work

2 Assumptions and Notations

2.1 Assumptions

Due to the lack of necessary data, we make the following assumptions to help us perform modeling:

1. Seven and more guesses in Wordle are considered to be 7 tries.
2. The change of operating entity has no effect on player participation.

2.2 Notations

Table 1 shows the necessary notations and signs used in this paper. Other notations and signs will be declared or defined when using.

Symbol	**Descripitions**
X_{1D}	the original 1D structure of time series
X_{2D}^{i}	2D tensors reshaped by the 1D time series
$X_{2D}^{l,i}$	a set of 2D tensors
R^2	the fitting degree of the fitting curve and gamma curve
I	information content
p	the probability of obtaining this information
$E(I)$	information entropy
r_s	Spearman correlation coefficient
F	Features fusion matrix
h	one hot code to convert duplicate count

Table 1: Notions and Symbol Description

3 Model I: Temporal 2D-Variation Model

In order to better predict the number of reported results in the future, we built a temporal 2D-variation model[3].

We find that there is no obvious periodicity of the number of reported results. At the same time, there are a lot of small fluctuations in the curve.

Secondly, we find out that the variation of each time point is not only affected by the temporal pattern of its adjacent area but also highly related to the variations of its adjacent periods. For clearness, we name these two types of temporal variations as intraperiod-variation and interperiod-variation respectively. The former indicates short-term temporal patterns within a period. The latter can reflect long-term trends of consecutive different periods.

These variations within and between periods are difficult to predict by the original 1D time series model, so we established a model to disentangle the intricate temporal patterns.

Before building the model, we processed the data. By drawing the box graph, we found some outliers and deleted them.

3.1 Overall Architecture

Step 1: Transform 1D into 2D T is the time series, that is, the time span in the data file, which is 359 days in total. C is recorded variates： number of reported results. The original 1D organization is $\mathbf{X}_{1\mathrm{D}} \in \mathbb{R}^{T \times C}$.

We reshaped the 1D time series into multiple 2D tensors by the following equations:

$$\mathbf{X}_{2\mathrm{D}}^{i} = \mathrm{Reshape}_{p_i, f_i}\left(\mathrm{Padding}\left(\mathbf{X}_{1\mathrm{D}}\right)\right), i \in \{1, \cdots, k\},$$

Where Padding $(\cdot)$ is to extend the time series by zeros along the temporal dimension. p_i and f_i represent the number of rows and columns of the transformed 2D tensors respectively. $\mathbf{X}_{2\mathrm{D}}^{i} \in \mathbb{R}^{p_i \times f_i \times C}$ denotes the i-th reshaped time series based on frequency f_i.

Step 2： Extract two-dimensional time series change representation

For 2D tensors $\left\{\mathbf{X}_{2D}^{l,1}, \mathbf{X}_{2D}^{l,2}, \ldots, \mathbf{X}_{2D}^{l,k}\right\}$, we use 2D convolution to extract information. Here, we select the classic Inception model:

$$\widehat{\mathbf{X}}_{2\mathrm{D}}^{l,i} = \mathrm{Inception}\left(\mathbf{X}_{2\mathrm{D}}^{l,i}\right)$$

Step 3: Transform 2D into 1D

For the extracted temporal features, we convert them back to 1D space for information aggregation:

$$\widehat{\mathbf{X}}_{1\mathrm{D}}^{l,i} = \mathrm{Trunc}\left(\mathrm{Reshape}_{1,(p_i \times f_i)}\left(\widehat{\mathbf{X}}_{2\mathrm{D}}^{l,i}\right)\right), i \in \{1, \cdots, k\}$$

$\widehat{\mathbf{X}}_{1\mathrm{D}}^{l,i} \in \mathbb{R}^{T \times d_{\text{model}}}$, $\mathrm{Trunc}(\cdot)$ indicates to remove the 0 added by the operation Padding $(\cdot)$ in step1.

Step 4: Adaptive fusion

Similar to the design in Autoformer, we got the one-dimensional representation $\left\{\widehat{\mathbf{X}}^{l,1}, \ldots, \widehat{\mathbf{X}}^{l,k}\right\}$. The intensity of the corresponding frequency is weighted and summed to obtain the final output.

$$\widehat{\mathbf{A}}_{f_1}^{l-1}, \cdots, \widehat{\mathbf{A}}_{f_k}^{l-1} = \mathrm{Softmax}\left(\mathbf{A}_{f_1}^{l-1}, \cdots, \mathbf{A}_{f_k}^{l-1}\right)$$

$$\mathbf{X}_{1\mathrm{D}}^{l} = \sum_{i=1}^{k} \widehat{\mathbf{A}}_{f_i}^{l-1} \times \widehat{\mathbf{X}}_{1\mathrm{D}}^{l,i}$$

Through step 1 to step 4, we got a Timesblock, as shown in Figure 2.

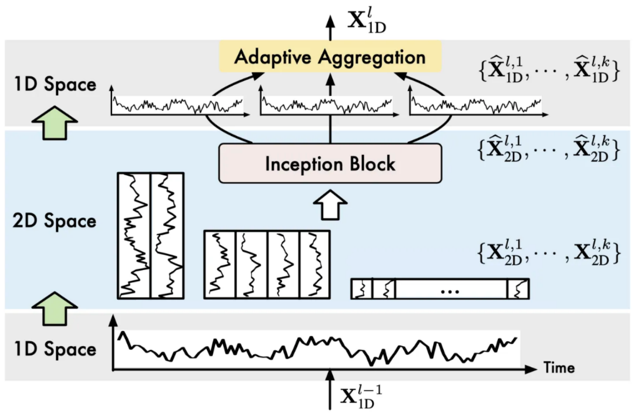

$\mathbf{X}_{1\mathrm{D}}^{l}$
Adaptive Aggregation
1D Space
$\{\widehat{\mathbf{X}}_{1\mathrm{D}}^{l,1}, \cdots, \widehat{\mathbf{X}}_{1\mathrm{D}}^{l,k}\}$
Inception Block
$\{\widehat{\mathbf{X}}_{2\mathrm{D}}^{l,1}, \cdots, \widehat{\mathbf{X}}_{2\mathrm{D}}^{l,k}\}$
2D Space
$\{\mathbf{X}_{2\mathrm{D}}^{l,1}, \cdots, \mathbf{X}_{2\mathrm{D}}^{l,k}\}$
1D Space
Time
$\mathbf{X}_{1\mathrm{D}}^{l-1}$

Figure 2: TimesBlock flowchart

The Temporal 2D-Variation Model consists of stacked TimesBlocks, as shown in Figure 3. The input sequence first passes through the embedding layer to obtain the depth feature $\mathbf{X}_{1\mathrm{D}}^{0} \in \mathbb{R}^{T \times d_{\text{model}}}$. For the layer l of TimesBlock, its input is $\mathbf{X}_{1\mathrm{D}}^{l-1} \in \mathbb{R}^{T \times d_{\text{model}}}$, and then 2D time series changes are extracted by 2D convolution:

$$\mathbf{X}_{1\mathrm{D}}^{l} = \mathrm{TimesBlock}\left(\mathbf{X}_{1\mathrm{D}}^{l-1}\right) + \mathbf{X}_{1\mathrm{D}}^{l-1}$$

3.2 The Fitting Curve Follows Gamma Distribution

We take T and the number of reported results as the main input, and words as covariates into the model. The output is the fitting and prediction curve, as

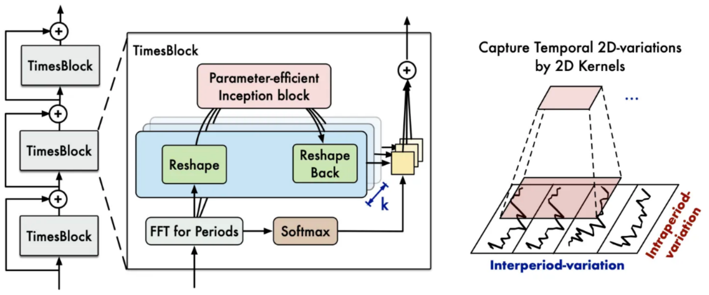


Figure 3: Aarchitecture of Temporal 2D-Variation Model

shown in Figure 4 and Figure 5.

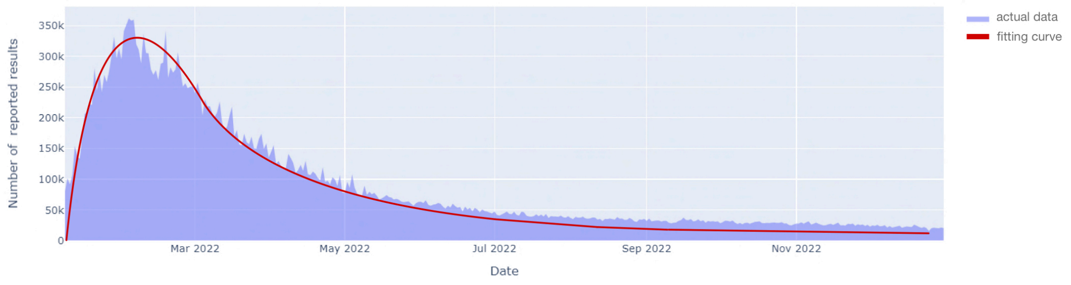


Figure 4: The number of reported results, The blue area is the actual quantity, and the red line is the curve fitted by the model.

We found that the fitting curve is very close to the gamma distribution:

$$\begin{cases} f(x) = \frac{(\alpha x)^{\beta-1}}{\Gamma(\beta)}\alpha^{\mathrm{e}-2x}, 0 \leqslant x < \infty \\ \Gamma(\beta) = \int_0^x t^{\beta-1}\mathrm{e}^{-t}\,\mathrm{d}t \end{cases}$$

Where α is the shape parameter; β is a scale parameter. We use the R-Squared to calculate the fitting degree of the fitting curve and gamma curve, in which α is about 2.3, and β is about 1.2. We got R^2=0.913678, thus, the curve's fitting effect is excellent.

By observing the data, we find that the curve has a general trend of rising sharply and then falling slowly. This change is in line with people's interest in novelty. However, the curve is not smooth, and there are many fluctuations in it. For example, we found that on December 25, the number of reported results decreased significantly compared with the previous two days. Therefore, the number of games reported might have a lot to do with the holidays.

3.3 Prediction Results of Time Series

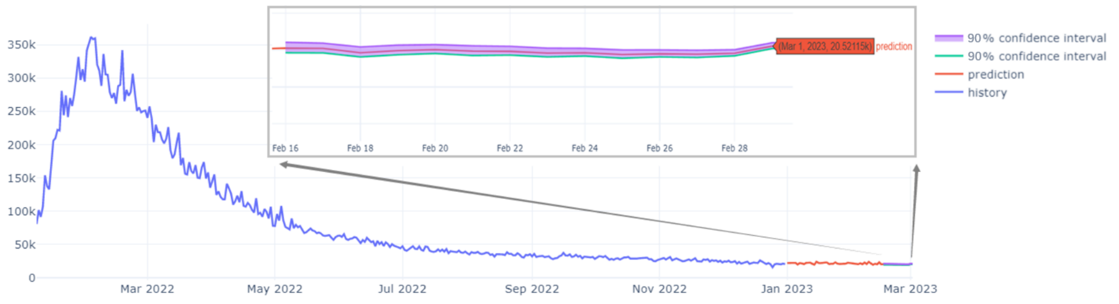


Figure 5: Prediction for the number of reported

The blue line is the historical data, and the red line is the result predicted by the Temporal 2D-Variation model. We use the arrow to enlarge the data of the last few days of the forecast, and the predicted data and confidence interval are in the box. In order to get the prediction interval for the number of reported results on March 1, 2023, we set a 90% confidence interval, which is the purple and green areas in the figure. The upper limit of the purple area is 20.98455K, and the lower limit of the green area is 20.19956K, **which means that if the range of the prediction result is between 20.19956K-20.98455K, the result will be at 90% confidence level**. The red line is the result predicted by the model.

3.4 Performance Analysis

In order to evaluate our Temporal 2D-Variation model, we selected classical models from both deep learning and statistical directions as a baseline and took several evaluation metrics for comparison, the results are as follows：

Model	**RMSE**	**MAPE**
ARIMA	0.7034	7.21
LSTM	0.6183	6.25
Temporal 2D-Variation	0.0342	0.36

Table 2: Model performance evaluation

By observing Table 2, it is not difficult to find that: the RMSE and MAPE of the Temporal 2D-Variation model are very close to 0. Its prediction accuracy is much higher than the prediction accuracy of traditional ARIMA or LSTM neural network models, and the prediction error is greatly reduced.

4 Quantification and Analysis of the Attributes of Words

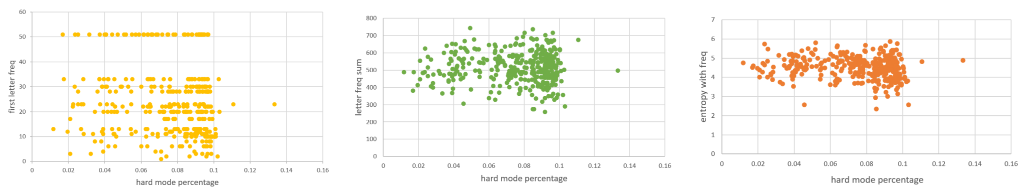
In this section, we developed a spearman correlation[4] test model to test whether any attributes of the word would have an effect on the percentage of participants in the hard mode.

4.1 Quantification of Word Attributes

First, we quantified the word attributes and presented the various attributes of words in numerical form. The scatter chart of attributes and percentage of scores reported is shown in Figure 6.

The five attributes of words we finally selected are as follows:

1. **Has duplicate:** judge whether there are duplicate letters in the word. If yes, set the value to 1. If no, set the value to 0.
2. **Duplicate count:** how many duplicate letters exist in a word.
3. **First letter freq:** frequency of occurrence of the first letter. The higher the frequency, the more common the letter is in the word, and vice versa.
4. **Letter freq sum:** the sum of the frequency of occurrence of all letters in the word.
5. **Entropy with freq:** weighted sum of entropy and frequency measures of words.



(a) the frequency corresponding to the first letter of that word (b) the sum of the frequencies of all letters in that word (c) weighted sum of information entropy and frequency

Figure 6: Each point represents the word of that day, the abscissa is hard mode percentage.

The process of calculating information entropy with freq is as follows:

The definition formula of information is: $I = log_2^{\frac{1}{p}}$, p is the probability of obtaining this information. we can calculate the expected value of the obtained information. This expected value is information entropy, set as $E(I)$.

$$E(I) = \sum p \times \log_2^{\frac{1}{p}}$$

Next, we define *freq* as the word frequency of each word. When we consider the rarity of a word, we need to consider the influence of information entropy and word frequency at the same time. So we add weights to these two impacts to better form the final weighted sum. If the weight of information entropy is w_1 and the weight of word frequency is w_2, then $w_1 + w_2 = 1$. The weighted sum formula is:

$$sum = E(I) * w_1 + freq * w_2$$

4.2 Results of Spearman Correlation Test Model

We quotient the number of people who choose the hard mode with the total number of people to find the percentage. Then we used the Spearman correlation coefficient to measure the strength and direction of the association between attributes of the word and the percentage.

$$r_s = 1 - \frac{6 \sum_{i=1}^{n} d_i^2}{n\left(n^2 - 1\right)}$$

Here,n is the number of data points of the two variables, and d_i is the difference in ranks of the "i-th" element. The Spearman Coefficient r_s can take a value between +1 to -1 where,

r_s value of +1 means a perfect association of rank.

r_s value of 0 means no association of ranks.

r_s value of -1 means a perfect negative association between ranks.

The closer the r_s value to 0, the weaker the association is between the two ranks.

Spearman	Has duplicate	Duplicate count	First letter freq	Letter freq sum	Entropy with freq
Hard mode percentage	0.083	0.084	-0.14	-0.023	-0.116

Table 3: Spearman correlation

It can be seen from the table 3 that the Spearman correlation coefficients of the five factors are very small. **So we can judge that the attributes of the word will not affect the percentage of scores reported that were played in Hard Mode**.

5 Model II: Convolutional Transformer with Word2Vec

5.1 Introduction to Transformer

A transformer[5] is a deep learning model that adopts the mechanism of self-attention, differentially weighting the significance of each part of the input data. We propose a multi-window Convolutional Transformer (ConvTransformer[6]) that takes the advantage of both the Transformer and CNN for regression.

The architecture of the regression model and the pipeline as shown in Figure 7.

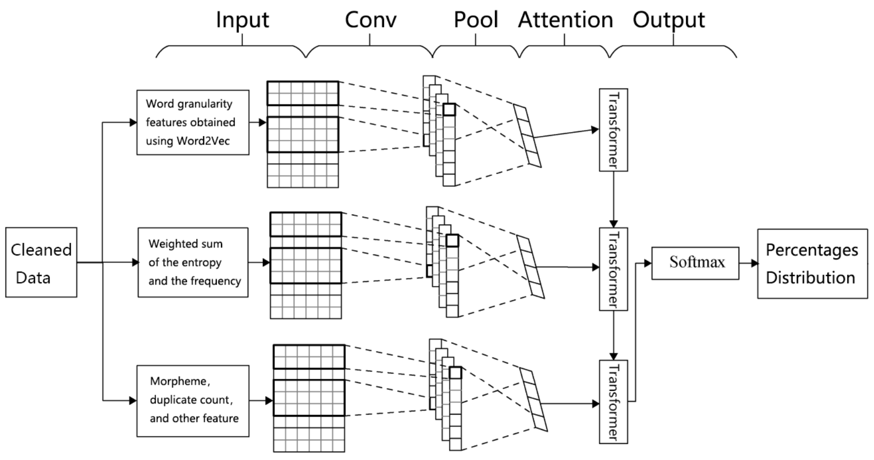

Input
Conv
Pool
Attention
Output
Cleaned Data
Word granularity features obtained using Word2Vec
Weighted sum of the entropy and the frequency
Morpheme, duplicate count, and other feature
Transformer
Transformer
Transformer
Softmax
Percentages Distribution

Figure 7: The architecture of Model 2

5.2 Feature Fusion

First of all, we have to extract features from the solution word, where we use **Word2Vec**[7]. Since the input is a single word, we need to use a model at the word granularity level. After obtaining the feature vectors of the words, we then stitch them together with the weighted sum of information entropy and frequency, the number of repeated letters, and the frequency of initial letters mentioned in the previous section, and finally we can input them into the neural network model. The specific calculation is shown in the flow chart of the algorithm.

Algorithm 1: Feature extraction and fusion.

```
Input: A given future solution word $w_i$ on a future date;
Output: Features fusion matrix $\mathbf{F} \in \mathbb{R}^{M \times N}$.
foreach word $w_i$ in $\mathcal{W}$ do
    Calculate word embedding vector $\mathbf{v}_i$ using pretrained word2vec;
    Convert duplicate count to one hot code $\mathbf{h}_i$
    Calculate the weighted sum of information entropy and frequency;
    Compute the sum of the frequency of each letter of the word $w_i$.
Update $\mathbf{F}$;
Fusion of multiple independent vectors into a single feature matrix $\mathbf{F}$ by PCA.
```

5.3 Methodology and Theory

In this section, we present our methodologies in detail. The input word is encoded using the proposed Convolutional Transformer (ConvTransformer) to capture local n-gram features; Meanwhile, attributes embedding is generated for each word utilizing an external knowledge base[8]; Then, feature-aware attention is used to summarize the output of ConvTransformer and obtain the final representation by incorporating entropy information and frequency information of each token; Finally, the distribution of the reported results is predicted based on the final representation.

Convolutional Transformer Encoder

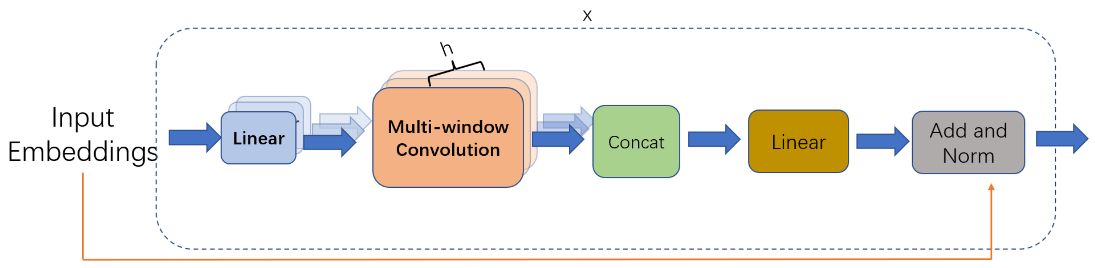


Figure 8: Convolutional Transformer (ConvTransformer)

1. **N-gram Convolution Operation**:

   Convolution is the fundamental operation of ConvTransformer which computes the semantic relevance between n-grams in the word and trainable convolutional filters. Specifically, given the input Words $W = [W_1, W_2, \ldots, W_l]$, each word W_i is represented as d_w-dimensional word embedding $\mathbf{x}_i$ by looking up the word embedding matrix $\mathbf{W}^{\text{wrd}} \in \mathbb{R}^{d_w \times V}$, where V is vocabulary size. Then, n-gram convolution over input embeddings $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_l]^T$ is performed using convolutional filters $\mathbf{F} = [\mathbf{f_1}, \mathbf{f_2}, \ldots, \mathbf{f_m}]^T$, where $\mathbf{f_i} \in \mathbb{R}^{nd_w}$ and n is the convolution window size. Consequently, a feature map $\mathbf{M} \in \mathbb{R}^{l \times m}$ is generated as follows:

$$\mathbf{M} = \mathbf{X} \otimes \mathbf{F}^T$$

Where $\circledast$ indicates the convolution operation of $\mathbf{f_i}$ over $\mathbf{X}$. Specifically, the value in the feature map is calculated as shown in Equation:

$$m_{ji} = f\left(\mathbf{f_i}^T \cdot \text{Cat}\left(x_j, x_{j+1}, \ldots, x_{j+n-1}\right) + b\right)$$

Where Cat means concatenation, f is a non-linear activation function and b is a bias term.

2. **Multi-head Multi-layer Structure**:

   Inspired by the structure of the Transformer, ConvTransformer also has a multi-head multi-layer structure that allows the model to extract features in both parallel and sequential manners. As shown in Figure 8, for a h-head ConvTransformer, the input embedding X is first transformed into h sub-spaces using different linear transforms. Then, different n-gram convolution is performed in different sub-spaces using multiple convolution window sizes. Finally, the outputs from all the convolutional heads are concatenated together and linearly transformed to the original input dimension, as shown in Equation:

$$\text{ConvTransformer}\left(\mathbf{X}\right) = \text{Cat}\left(\mathbf{M}_1, \mathbf{M}_2, \ldots, \mathbf{M}_h\right) \mathbf{W}^M$$
$$\text{where } \mathbf{M}_i = \text{Conv}\left(\mathbf{X}\mathbf{W}_i^X\right)$$

   Here, Cat indicates column-wise concatenation, $Conv$ indicates n-gram convolution operation, $\mathbf{W}_i^X \in \mathbb{R}^{d_w \times (d_w/h)}$ and $\mathbf{W}^M \in \mathbb{R}^{hm \times d_w} \mathbf{W}^M \in \mathrm{R}^{hm \times dw}$ are the weight matrices of linear transformations. Moreover, we adopt the residual connection and layer norm.

Feature-aware Attention

To effectively incorporate the entropy information and frequency information of attributes embedding and obtain the final word representation, we propose a feature-aware attention mechanism that summarizes the output of ConvTransformer while taking both feature and time information into consideration. The attention weight of each token α_i is calculated based on its feature embedding $\mathbf{s}_i$ and time embedding $\mathbf{t}_i$ besides the ConvTransformer output $\mathbf{o}_i$. The time embedding $\mathbf{t}_i$ is obtained by mapping the token's absolute time to d_p-dimensional embeddings based on a trainable time embedding matrix $\mathbf{W}^t \in \mathbb{R}^{d_t \times T}$, where T is the total number of times.

Formally, the attention weight α_i is calculated as follows:

$$u_i = \mathbf{c}^{\mathrm{T}} \tanh\left(\mathbf{W}_o \mathbf{o}_i + \mathbf{W}_s \mathbf{s}_i + \mathbf{W}_t \mathbf{t}_i\right)$$
$$\alpha_i = \frac{\exp\left(u_i\right)}{\sum_{j=1}^{l} \exp\left(u_j\right)}$$

Where $\mathbf{W}_o \in \mathbb{R}^{d_a \times d_w}, \mathbf{W}_s \in \mathbb{R}^{d_a \times 2d_s}, \mathbf{W}_t \in \mathbb{R}^{d_a \times d_t}$, d_a is attention dimension, and $\mathbf{c} \in \mathrm{R}^{da}$ is a word vector learned by the neural network. The attention weight α_i reflects the contribution of each token to the final word representation, and the final representation is computed as shown in Equation:

$$\mathrm{f} = \sum_{i=1}^{l} \alpha_i \mathbf{O}_i$$

The feature embedding used in our feature-aware attention mechanism contains prior knowledge of words' attributes, it is beneficial for regression and also reduces the reliance on training data; The time embedding used in feature-aware attention provides time information of each token, which is important for regression since words appear in different days may have different affective meanings.

Regression and Optimization

After obtaining the final word representation f, we pass it to a regressor to compute the distribution of the reported results. The regressor consists of a fully connected layer of standard neural network for dimension reduction and a softmax layer (fully connected layer with softmax activation function). For optimization, we train our model by minimizing categorical cross-entropy loss and center loss using mini-batch stochastic gradient descent (SGD) with momentum. We also apply dropout regularization before feeding f into the regressor to prevent co-adaptation of hidden units.

5.4 Prediction of the Distribution of the Reported Results

In the Ubuntu conda environment, We've trained the ConvTransformer for

100 epochs, with two RTX 3090 graphics cards for acceleration.

The evaluation metric we used is MAE. Results show that our model achieves the best performance on our Wordle datasets, the MAE we finally got is 0.4598, **take “EERIE” as an example, the predictions are as follows**:

Percentages	**1 try**	**2 tries**	**3 tries**	**4 tries**	**5 tries**	**6 tries**	**Failed**
“EERIE”	0	1.7	9.3	27.9	34.7	21.7	6.3

Table 4: Prediction of the distribution

5.5 The Confidence of the Model and the Prediction Uncertainty

With our forecasting model, there are at least four sources of uncertainty:

- The random error term
- The parameter estimates
- The choice of model for the historical data
- The continuation of the historical data-generating process into the future

The ConvTransformer, with its excellent performance, has been trained to significantly weaken the effects of random errors and parameter estimation. **That is to say, the model uncertainty and the DGP uncertainty are mainly associated with our model and predictions**.

To solve the above problem, our prediction uses quantile regression. Quantile regression aims at estimating the conditional quantiles of the response variable. For a continuous distribution function, the α-quantile $Q_\alpha(x)$ is defined such that the probability of Y being smaller than $Q_\alpha(x)$ is, for a given $X = x$, equal to α.

Therefore, a specific loss function is needed in order to train models that predict quantiles. The metric we used for quantile regression is called pinball loss[9]:

$$\text{pinball}(y, \hat{y}) = \frac{1}{n_{\text{samples}}} \sum_{i=0}^{n_{\text{samples}}\,1} \alpha \max(y_i - \hat{y}_i, 0) + (1-\alpha) \max(\hat{y}_i - y_i, 0)$$

where α is the target quantile, y the real value and $\hat{y}$ the quantile prediction.

All in all, with the supervision of this strategy, our prediction results have a confidence level of 95% .

6 Difficulty Classification Based on Model II and Statistics

On the basis of Model II, we made some improvements to make the model available for classification. The workflow of the model is shown in Figure 9.

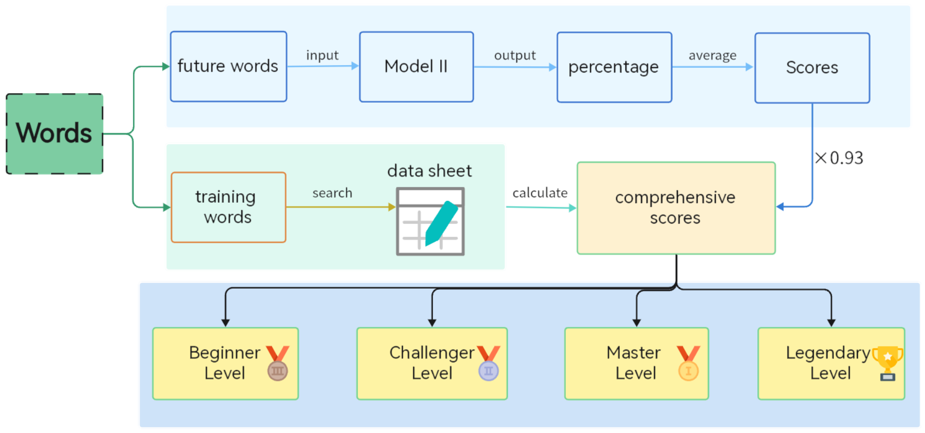


Figure 9: The workflow of the model

6.1 Classification Basis

We calculated the comprehensive score of each word and ranked the scores to obtain the difficulty. **First**, we calculate the average number of guesses for each word according to the percentage that guessed the word. **Secondly**, we found that two words with the same average guess times do not necessarily have the same difficulty, because the proportion of people who choose the difficult mode on the day is different. Obviously, if word A is more difficult to choose than word B, but both A and B get the same score, then it can be proved that word A is simpler than word B. **Therefore**, in order to better evaluate the difficulty of words, we take into account the proportion of people who choose the difficult mode. **Finally**, a comprehensive score is calculated. The specific calculation method is:

$$Scores = try\_num * simple\_per$$

$Scores$ is the comprehensive score of each word, try_num is the average number of guesses of the word, and simple_per is the proportion of people who choose the simple mode.

We counted the comprehensive scores of all words. The drawn violin plot is as shown in Figure 10.

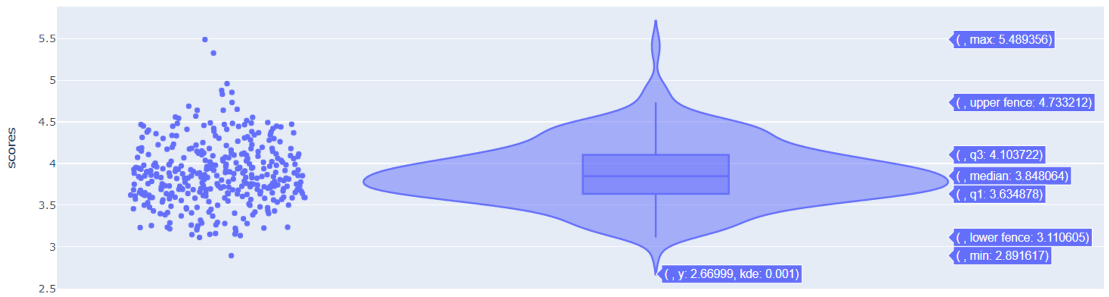


Figure 10: Violin plot of comprehensive scores

$q1$ and $q3$ are the quartiles in the comprehensive scores. We use $q1, median, q3$ as the dividing nodes to divide the word difficulty into four levels, as shown in the table 5.

Level	**Beginner Level**	**Challenger Level**	**Master Level**	**Legendary Level**
Scores	<3.63	3.63-3.84	3.84-4.10	>4.10

Table 5: Wordle Difficulty Grading

6.2 Weight Acquisition Based on Gradient Descent Algorithm

In order to get the weight of each word attribute, we adopt the stochastic gradient descent(SGD) with momentum[10].

The formula for calculating the gradient and updating parameters is as follows:

Algorithm 2: SGD with momentum.

```
Input: learning rate $\boldsymbol{\eta}$; momentum parameter $\boldsymbol{\alpha}$;
       initial parameter $\boldsymbol{\Theta}$;initial velocity $\mathbf{v}$.
while not yet converged do
    Select $m$ samples from the training set,$\left\{\mathbf{x}^{(1)}, \cdots, \mathbf{x}^{(n)}\right\}$, where the
     target corresponding to $\mathbf{x}^{(i)}$ is $\mathbf{y}^{(i)}$;
    Gradient calculation: $\mathbf{g} \leftarrow \nabla_{\Theta} \sum_i L\left(f\left(\mathbf{x}^{(i)}; \Theta\right), \mathbf{y}^{(i)}\right)/m$ ;
    Velocity Update: $\mathbf{v} \leftarrow \alpha \mathbf{v} - \eta \mathbf{g}$;
    Parameter Update: $\Theta \leftarrow \Theta + \mathbf{v}$.
```

Through the above method, we get the corresponding weights of the five attributes of words. As shown in Figure 11. the red line represents the highest priority, representing the largest weight of this attribute, and the blue line represents the lowest priority.

Our conclusion is that duplicate count has the largest weight among all attributes and has the greatest impact on the results, and letter_ freq_sum has the least weight and has the least impact on the results.

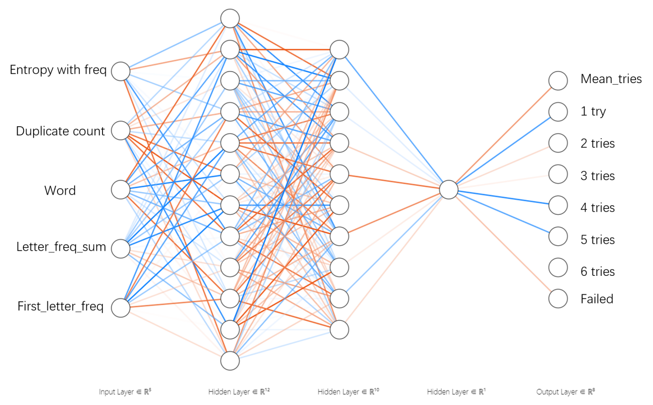


Figure 11: Weight of words attribute, the more red lines are, the more important the attribute is. The red line is the opposite.

6.3 Classification Results and Performance of the Model

The Wordle dataset has 359 data, we divided it into a training set, validation set, and test set according to 3:1:1. After adding labels according to the difficulty partition described in the previous section, we use ConvTransformer to train. The accuracy variation curve of the final training set is as follows:

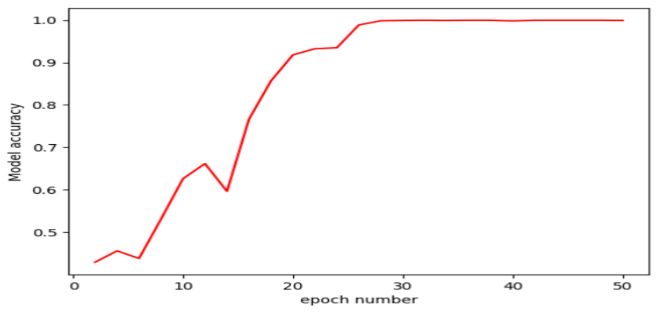


Figure 12: Training set accuracy variation curve

Metric	Accuracy	F1-score
Valid	95.4%	0.82
Test	93.9%	0.88

Table 6: Classification Performance

We use accuracy and F1-score to measure the classification performance, and we can see that the results are excellent in both the validation and test sets.

Again, using "EERIE" as an example, we can calculate its score to be 4.1158. Based on the results of the table 5, we consider it to be at the legendary level of difficulty, which is very much in line with our guess and has an accuracy rate of almost 94%.

7 Some Interesting Features

After analyzing the data, we obtained several new pieces of information as follows.

7.1 Difficult Mode Number Ratio

We calculated the proportion of people who choose the hard mode in the total number and plotted the trend change of the proportion(Figure 13).

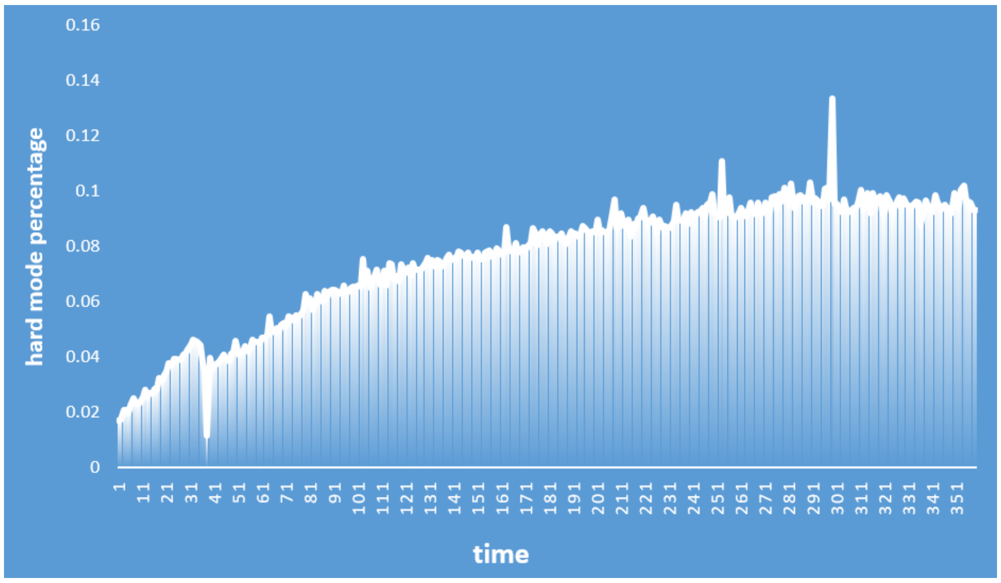


Figure 13: Difficult mode headcount ratio over time

We can find that as time goes on, the proportion of players choosing the hard mode is gradually increasing. After analysis, we believe that this is most likely due to the fact that players increasingly want more challenge in their play.

Another possibility is that the strength of players is rising, and the easy mode is becoming increasingly difficult to meet the needs of players for a challenge.

Or maybe developers are constantly improving the game experience in the difficult mode so that the hard mode is bringing more interest to people.

7.2 Festival

We believe that certain special factors, such as holidays, may affect the feedback data. On holiday days, players may not devote their time to the game. For this reason, we have collected data for certain holidays and the day before them throughout the year in the United States.

Date	Number of players the day before	Number of players on the day
2022.9.5	32018	32733
2022.10.10	28408	26878
2022.11.11	27467	25993
2022.11.25	27705	24197
2022.12.25	20281	15554

Table 7: Player data for certain holidays and the day before them

The data in table 7 shows that the number of participants in the game decreases on most holiday days. The largest rate of decline is on Christmas Day (12.25), so we can believe that the number of players drops during the holidays due to the time of year. Also, we can guess that other factors may also have a short-lived effect on the number of people playing the game, not just the heat.

7.3 Initial

We found that there is a certain pattern in the frequency of occurrence of the initial letters of words. This frequency of occurrence of initial letters may affect the direction of players' word guesses, the difficulty of the words, and the average score of players. We investigated the initial letters of words in the

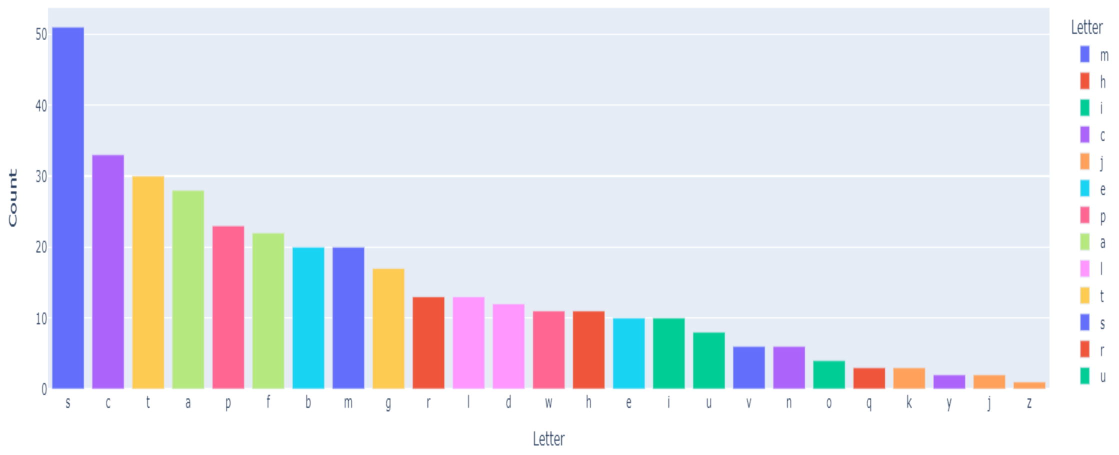


Figure 14: Frequency of initials

dataset.

We found that the number of words starting with S is very huge, while the number of words starting with J, Y, and Z is very small. Therefore, we can assume that we should try to guess words starting with S. This is because such words occur more frequently. We also guessed that the difference in the beginning letters might cause a change in the average score. In turn, it also affects the difficulty of the word. We made statistics for the letters beginning with S, J, Y, and Z(Figure 14).

Initial	**Average tries**
All words	4.19
s	4.10
j	4.60
y	4.48
z	4.79

According to the table, the average score of letters beginning with S is low, which shows that players can do better in the words beginning with S. The average score of words starting with uncommon letters such as J, Y, and Z is high, indicating that it is difficult for players to guess such words.

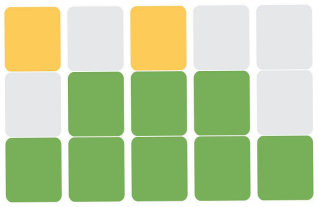
8 Evaluation of Strengths and Weaknesses

8.1 Strengths

- Our model is robust under the sensitivity tests, indicating that small changes in parameters won't lead to huge differences in results.
- The one-dimensional time-series data is extended to two-dimensional space for analysis, enabling unified modeling of intra-cycle and inter-cycle variation.
- It is not clear whether the difficulty mode ratio and word attributes are likely to be nonlinearly correlated.The spearman correlation test model can help us analyze their nonlinear relationship.
- Combines the advantages of convolution and deformer to show the regression performance and classification performance of the model efficiently and accurately.

8.2 Weaknesses

- The long-term prediction of our model may not be precise enough, considering the fact that we don't have long-term data to validate the long-term predictions.
- Feature vector extraction of a single word is limited by the performance of pre-trained Word2Vec.
- To simplify our model, we have made some assumptions that are not close enough to reality, which would also lead to errors in the results.



From: Team 2309900 , MCM C
To: The Puzzle Editor of the New York Times
Date: February 19, 2023

Dear Editor:

I am pleased to present to you an analysis of Wordle, which has become a global phenomenon in 2022. Based on the data you provided, we have developed two models that accurately predict Wordle's development trends and evaluate the difficulty of each word, enabling you to make corresponding strategic adjustments.

First, we developed the Temporal 2D-Variation Model, which predicts that the number of players on March 1st will be between 20.2K-21.0K. Additionally, we conducted a Spearman correlation analysis on a series of indicators and found that the attributes of the word will not affect the percentage of scores reported that were played in Hard Mode.

Second, we developed a model called ConvTransformer, which used quantile regression to estimate the percentage of tries, and our model has a 95% confidence level.

Third, we improved upon the second model by building a classification model using SGDM to obtain attribute weights for each word. We divided the words into four categories based on the overall scores, and we achieved an accuracy of 94% as measured by F1-score.

Finally, our analysis of the dataset revealed some interesting findings, such as an increasing proportion of players choosing the hard mode, the impact of holidays on the number of players, and the fact that words beginning with the letter "S" are more likely to be guessed correctly.

To make Wordle even more popular, we suggest the following based on our analysis:

- You can utilize our classification model to divide the difficulty into four levels, each corresponding to a distinct vocabulary. Players are free to choose the level they prefer.
- The sharing of everyone's achievements has greatly contributed to the widespread popularity of Wordle. To reward players for their efforts, experience points can be assigned based on the comprehensive performance of players.
- You may select an interesting answer process from the daily Twitter score sharing and feature it in the Wordle module of the *New York Times*.

Sincerely hope that Wordle can have better development!

Sincerely,
MCM C Team 2309900

References

[1] Wordle - The New York Times. (2023). [Online]. Available: https://www.nytimes.com/games/wordle/index.html

[2] Wordle Stats. (2023). [Online]. Available: https://twitter.com/WordleStats

[3] H. Wu, T. Hu, Y. Liu, H. Zhou, J. Wang, and M. Long, "Timesnet: Temporal 2d-variation modeling for general time series analysis," 2022. [Online]. Available: https://arxiv.org/abs/2210.02186

[4] F. Smarandache, "Alternatives to pearson's and spearman's correlation coefficients," 2008. [Online]. Available: https://arxiv.org/abs/0805.0383

[5] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2017. [Online]. Available: https://arxiv.org/abs/1706.03762

[6] P. Li, P. Zhong, J. Zhang, and K. Mao, "Convolutional transformer with sentiment-aware attention for sentiment analysis," in *2020 International Joint Conference on Neural Networks (IJCNN)*, 2020, pp. 1–8.

[7] Y. Goldberg and O. Levy, "word2vec explained: deriving mikolov et al.'s negative-sampling word-embedding method," 2014. [Online]. Available: https://arxiv.org/abs/1402.3722

[8] English Word List ——Lexipedia. (2023). [Online]. Available: https://en.lexipedia.org/

[9] G. Biau and B. Patra, "Sequential quantile prediction of time series," 2009. [Online]. Available: https://arxiv.org/abs/0908.2503

[10] T. M. Breuel, "On the convergence of sgd training of neural networks," 2015. [Online]. Available: https://arxiv.org/abs/1508.02790