Scoring reviews

Daniel Saharov, Andrey Shinkarenko, Kirill Jukov

May 2025

Abstract

This application analyzes arrays of restaurant reviews and categorizes them into key food-related groups such as meat and fish. Each category is assigned a score indicating its relevance within the reviews, providing a clear and quantitative overview of customer preferences. Github link: `https://github.com/Pifocoder/rating-reviews`.

1 Introduction

Analyzing customer reviews is a critical component in understanding consumer preferences and improving service quality in the restaurant industry. However, extracting meaningful insights from large volumes of unstructured text data remains a challenging task. This paper presents an application designed to process arrays of restaurant reviews and automatically categorize them into predefined food-related categories such as meat, fish, and others. Each category is assigned a relevance score that reflects its prominence within the reviews. By leveraging natural language processing techniques, the system provides a quantitative summary of customer feedback, enabling data-driven decision-making for restaurateurs and researchers. This approach contributes to the growing field of opinion mining and sentiment analysis by focusing on domain-specific categorization with scoring, facilitating more targeted and actionable insights.

1.1 Team

Daniel Saharov train model for scoring reviews
Andrey Shinkarenko develop backend and find keywords
Kirill Jukov testing application

2 Related Work

The task of extracting meaningful insights from restaurant reviews has been extensively studied, with a focus on sentiment analysis and keyword extraction techniques. Sentiment analysis methods often employ natural language processing (NLP) to evaluate the emotional tone of reviews, enabling businesses to gauge customer satisfaction and identify areas for improvement [?]. These approaches typically involve data collection, text preprocessing, feature extraction, and classification using machine learning models such as Support Vector Machines or neural networks [?, ?].

Aspect-based sentiment analysis (ABSA) has gained popularity for its ability to associate sentiments with specific attributes like food quality, service, and ambiance, providing more granular insights [?]. Unsupervised learning algorithms have also been applied to restaurant review sentiment classification, addressing the scarcity of labeled data [?]. These methods leverage lexicon-based features and clustering techniques to detect sentiment patterns without extensive training datasets.

Keyword extraction, a complementary task, is crucial for identifying main topics discussed in reviews. Unsupervised methods such as YAKE have been employed to extract relevant keywords without requiring annotated corpora, making them suitable for domain-specific applications like restaurant reviews [2, 3]. Such approaches rely on statistical features including term frequency and position to score candidate keywords effectively.

Recent works also explore the integration of sentiment analysis with keyword extraction to provide a comprehensive understanding of customer feedback [1]. This combined approach facilitates the identification of key categories (e.g., meat, fish) and their associated sentiment scores, enabling actionable insights for restaurateurs.

Overall, the existing literature demonstrates a trend towards leveraging unsupervised and hybrid methods to overcome data limitations while enhancing the interpretability and usefulness of review analysis. Our work builds upon these foundations by developing an application that categorizes restaurant reviews into food-related categories with associated relevance scores, employing both keyword extraction and sentiment evaluation techniques.

3 Model Description

This approach tackles review classification using a two-stage training process leveraging a pre-trained BERT-based model. The core idea is to first learn robust and discriminative text representations through self-supervised contrastive learning and then fine-tune these representations for the specific task of sentiment classification. **Model Architecture** The architecture of the model is designed to be flexible for both the self-supervised pre-training and supervised fine-tuning stages:

1. BERT Backbone:

    - Utilizes a pre-trained transformer model (e.g., cointegrated/rubert-tiny2 or RussianNLP/ruRoBERTa-large-rucola) as its core. This backbone is responsible for generating contextualized embeddings for the input review text.

- The output from the [CLS] token of the BERT model's last hidden state is used as the primary representation of the entire review.

2. Projection Head:
   - This is a Multi-Layer Perceptron (MLP) connected to the BERT [CLS] token output. Its structure is:
     - `Linear(bert_hidden_size, 256)`
     - `ReLU()`
     - `Linear(256, 192)`
     - `LayerNorm(192)`
   - Purpose: In the context of contrastive learning (Phase 1), the projection head maps the BERT representations into a new latent space where the contrastive loss is applied. This is a common practice as it has been shown that applying contrastive loss directly on BERT [CLS] outputs can sometimes be less effective than on features from a non-linear projection. For the classification phase (Phase 2), these projected features serve as enriched input to the classifier. Kaiming Normal initialization is used for its linear layers.
3. Classifier Head (Conditional)
   - `This head is only active when use_classifier=True (during phase 2). Its structure is:`
     - `Linear(192, 128)`
     - `Dropout(dropout_rate)`
     - `Linear(128, num_classes)`
   - Purpose: This is a standard classification layer that takes the (potentially refined) 192-dimensional features from the projection head and maps them to the final output logits for each review class (e.g., 1 to 5 stars). Xavier Normal initialization is used for its linear layers

Why this Architecture?

- **Leveraging Pre-trained Knowledge** `Using a BERT backbone allows the model to benefit from the vast amount of linguistic knowledge learned during its initial pre-training on large text corpora.`
- **Feature Refinement** `It can help in disentangling features and improving representation quality.`
- **Modularity** `The separation of the projection head and classifier head allows for a clean two-stage training process. The same core architecture (BERT + projection head) can produce features for contrastive learning and then be seamlessly extended with a classifier for the downstream task.`

Two-Stage Training Process The model is trained in two distinct phases:

- **Phase 1: Self-Supervised Contrastive Pre-training:**
  - **Objective:** To learn rich and discriminative text representations without relying on explicit class labels.
  - **Mechanism:** The model (BERT + projection head) processes original and augmented versions of reviews. A contrastive loss function (incorporating elements of SimCLR, supervised contrastive loss, K-Means style loss, and cluster contrastive loss) is applied to the outputs of the projection head. This loss encourages representations of semantically similar reviews (e.g., an original review and its augmentation, or reviews from the same (pseudo) class) to be close to each other in the embedding space, while pushing dissimilar ones apart.
  - **Benefits:**
    - Better Representations: Leads to embeddings that capture more nuanced semantic information relevant to distinguishing between different types of reviews.
    - Improved Generalization: Pre-training on a related self-supervised task can help the model generalize better to the downstream classification task, especially with limited labeled data for classification.
- **Phase 2: Supervised Fine-tuning for Classification:**
  - **Objective:** o adapt the learned representations for the specific task of classifying reviews into predefined sentiment categories.
  - **Mechanism:** The BERT backbone and the projection head (with weights initialized from Phase 1) are now connected to the classifier head. The entire model (or parts of it, see "Weight Freezing") is then fine-tuned using a standard classification loss (e.g., CrossEntropyLoss) on labeled review data.
  - **Benefits:**
    - Faster Convergence: Starting with well-initialized weights from Phase 1 often leads to faster convergence during supervised fine-tuning.
    - Higher Performance: The refined representations from Phase 1 can lead to better classification accuracy and F1-scores compared to training a classifier directly from a standard pre-trained BERT.

Logic of Loss Calculation

The cornerstone of the self-supervised pre-training phase (Phase 1) is the `TotalContrastiveLoss` module. This module synergistically combines three distinct loss functions, each designed to foster the learning of robust, well-structured, and discriminative representations for review texts. The final objective is a weighted sum of these individual loss components, allowing for a nuanced approach to representation learning.

The core of our self-supervised pre-training strategy (Phase 1) revolves around the `TotalContrastiveLoss` module. This module integrates three distinct loss functions, each designed to impart specific desirable properties to the learned representations of review texts. By optimizing a weighted sum of these components, we aim to produce feature embeddings that are robust, well-structured, and highly discriminative, thereby providing a strong foundation for the downstream classification task.

Instance-Level Contrastive Loss ($\mathcal{L}_{inst}$)

Core Idea: This component focuses on refining the local structure of the embedding space. It endeavors to make the learned representation (embedding) of an original review, denoted h_{orig}, more similar to the representation of its augmented version, h_{aug}. Concurrently, it pushes these paired representations further apart from the embeddings of all other reviews within the processed batch.

Mechanism: The loss operates on a batch of original feature vectors h_{orig} and their corresponding augmentations h_aug.

- Feature vectors are typically L2-normalized, ensuring that subsequent similarity calculations emphasize the cosine similarity (i.e., vector orientation) rather than vector magnitude.

- A pairwise similarity matrix is computed across all features (both h_{orig} and h_{aug}) in the batch. These similarities are then scaled by an instance-level temperature parameter, τ_{inst}, which controls the sharpness of the resulting probability distribution over pairs.

- **Supervised Variant (SupCon):** If class labels are available and the `use_supervised_instance_loss` flag is true, the definition of "positive" pairs is broadened. In this mode, positive pairs include not only an instance and its augmentation but also any two instances (original or augmented) that share the same class label. The objective thus becomes to attract all instances of the same class towards a common region in the embedding space, while simultaneously repelling instances from different classes.

- **Unsupervised Variant (SimCLR-style):** In the absence of class labels or if `use_supervised_instance_loss` is false, positive pairs are strictly defined as an original instance $h_{orig,i}$ and its corresponding augmentation $h_{aug,i}$. The loss encourages the augmented view of an instance to be more similar to its original view than to any other instance (original or augmented) present in the batch.

Goal: The primary aim of $\mathcal{L}_{inst}$ is to learn fine-grained, instance-level discrimination. For review analysis, this implies that the model learns to identify the nuanced characteristics that define a specific review's uniqueness or its semantic equivalence to slight variations. When integrated with supervised labels (SupCon), it further learns to group reviews exhibiting similar sentiment or belonging to the same predefined class.

K-Means Style Loss / Cluster Distribution Loss ($\mathcal{L}_{cd}$)

Core Idea: Drawing inspiration from spectral clustering methodologies and the L_{cd} term proposed in the Deep Temporal Contrastive Clustering (DTCC) paper, this loss component encourages the learned feature embeddings $Z \in \mathbb{R}^{N \times D}$ (where N is the batch size and D is the feature dimension, representing either original or augmented views) to be inherently "clusterable." The objective is to guide the model towards producing features that naturally form tight and well-separated clusters.

Mechanism: This loss takes as input the feature representations Z (e.g., detached versions of h_{orig} or h_{aug}) and a cluster assignment matrix $Q \in \mathbb{R}^{N \times K}$ (where K is the number of target clusters). The matrix Q typically represents a soft assignment of each instance to one of the K clusters and is commonly derived from the top-K left singular vectors of Z, obtained via Singular Value Decomposition (SVD). The loss is formulated as:

$$\mathcal{L}_{cd}(Z, Q) = \text{Tr}(Z^T Z) - \text{Tr}(Q^T Z^T Z Q) \tag{1}$$

Minimizing this objective effectively seeks to maximize the inter-cluster variance while minimizing intra-cluster variance, framed within the mathematics of matrix traces and spectral properties. This makes the learned data representations more amenable to standard clustering algorithms like k-means.

Goal: $\mathcal{L}_{cd}$ enforces a global clustering structure upon the learned representations, thereby enhancing their organization and separability into distinct, meaningful groups.

Cluster-Level Contrastive Loss with Entropy ($\mathcal{L}_{cc}$)

Core Idea: This loss operates directly on the cluster assignment matrices, Q_{orig} and Q_{aug}, derived from the original and augmented views of the data, respectively. Its primary aim is to ensure that the high-level clustering structure inferred from the original data view remains consistent with the structure inferred from its augmented counterpart.

Mechanism:

- The loss considers the cluster assignment matrices Q_{orig} and Q_{aug}. The columns of these Q matrices (or, equivalently, the rows of Q^T) can be interpreted as representations of the K clusters themselves.
- A contrastive loss, scaled by a cluster-level temperature parameter τ_c, is applied to these cluster representations. The k-th cluster representation

derived from Q_{orig} is treated as a positive pair with the k-th cluster representation from Q_{aug}. These are encouraged to be similar, while being pushed away from other non-corresponding cluster representations (e.g., the j-th cluster, where $j \neq k$).

- **Entropy Term ($H(q)$):** To prevent trivial solutions, such as all data instances collapsing into a single cluster, an entropy regularization term is incorporated. This term, often denoted as $H(q)$ or formulated as an 'ne$_loss$'$(e.g.,$log$(K) - H(q))$, aims to maximize the entropy of the average cluster assignment probabilities $P(q_j)$. This encourages a more uniform and diverse distribution of instances among the available K clusters.

Goal: $\mathcal{L}_{cc}$ promotes consistency in the macroscopic cluster structure across different views of the data. It also plays a crucial role in preventing degenerate clustering solutions, ensuring that the learned groupings are non-trivial and informative.

Final Combined Loss

The `forward` method of the `TotalContrastiveLoss` module computes each of the aforementioned individual loss components ($\mathcal{L}_{inst}$, $\mathcal{L}_{cd}$, and $\mathcal{L}_{cc}$) and then combines them into a single objective function. This is achieved through a weighted summation:

$$\mathcal{L}_{total} = \lambda_{inst}\mathcal{L}_{inst} + \lambda_{cd}\mathcal{L}_{cd} + \lambda_{cc}\mathcal{L}_{cc} \quad (2)$$

The hyperparameters λ_{inst}, λ_{cd}, and λ_{cc} serve as weighting factors that control the relative influence of each loss term on the overall training objective and gradient updates.

Advantages of the Combined Loss Approach

The utilization of this multi-component loss function offers several distinct advantages for learning high-quality representations:

- **Multi-Level Representation Learning:** The approach simultaneously refines representations at different granularities. $\mathcal{L}_{inst}$ captures fine-grained, local similarities and differences, crucial for distinguishing individual reviews. In contrast, $\mathcal{L}_{cd}$ and $\mathcal{L}_{cc}$ operate at a more global, structural level, encouraging the formation of coherent themes or groups among the reviews. This synergy allows the model to learn representations that are both discriminative at the instance level and well-organized at the cluster/group level.

- **Improved Robustness and Generalization:** By learning from diverse perspectives (instance similarity, inherent clusterability, and cross-view cluster consistency), the model is conditioned to develop more robust features that are less susceptible to minor input variations. The self-supervised nature of these losses, particularly when operating without strong label guidance, facilitates the discovery of intrinsic data structures, often leading to enhanced generalization capabilities on downstream tasks like sentiment classification.

- **Enhanced Feature Quality for Downstream Tasks:** The primary objective of employing this sophisticated loss strategy in Phase 1 is the generation of superior quality feature embeddings. When these pre-trained embeddings are subsequently fed into a relatively simple linear classifier during Phase 2, they are expected to yield improved classification performance compared to training a classifier directly on standard BERT outputs. The features are effectively "pre-conditioned" to be more separable and semantically richer.
- **Better Handling of Unlabeled or Weakly Labeled Data:** While the supervised variant of $\mathcal{L}_{inst}$ (SupCon) can leverage available labels, the $\mathcal{L}_{cd}$ and $\mathcal{L}_{cc}$ components are fundamentally unsupervised. This makes the overall approach potent even when high-quality, extensive labels are scarce, as significant structural information can still be extracted from the data itself.
- **Structured Latent Space:** The cumulative effect of these losses is the creation of a well-structured latent (embedding) space. In this space, reviews that share similar underlying sentiment, topics, or other semantic properties are naturally grouped together. This intrinsic organization simplifies the subsequent classification task for the Phase 2 classifier.

In essence, the `TotalContrastiveLoss` aims to transcend simple instance-level discrimination by actively fostering a globally consistent and clusterable architecture within the learned representation space, reflecting a powerful paradigm in modern self-supervised learning.

4 Dataset

Dataset Description

The primary dataset utilized for this study consists of textual reviews, presumably sourced from an e-commerce platform or a review aggregation service. Each entry in the dataset is expected to contain at least two key pieces of information:

1. **Review Text**: The raw textual content of the user-generated review. This text is the primary input to our models.
2. **Rating/Sentiment Label**: A corresponding label indicating the sentiment or rating score associated with the review (e.g., a star rating from 1 to 5. These labels are crucial for the supervised fine-tuning phase (Phase

2) and can also be optionally leveraged during the self-supervised pre-training (Phase 1) if a supervised contrastive loss component is employed.

The dataset may undergo initial preprocessing steps such as filtering based on text length (e.g., removing excessively short or long reviews) to ensure data quality and manage computational load. The distribution of rating labels is also an important characteristic, as class imbalance can affect model training and evaluation.

Data Augmentation Strategy

To facilitate contrastive learning, which relies on comparing different "views" of the same data instance, a data augmentation strategy is employed. For textual data, a common and effective technique is **back-translation**.

Keywords finding

We employ YAKE for keyword extraction in reviews due to the lack of available supervised datasets for this specific task. YAKE (Yet Another Keyword Extractor) offers an unsupervised, language-independent approach to extract significant keywords directly from individual documents. By leveraging statistical features such as term frequency, position, and contextual relatedness, YAKE efficiently identifies the main words that best represent the content without requiring external corpora or training data. This makes YAKE a versatile and practical tool for various text mining applications. Since our goal is to extract only nouns as keywords, we address this by performing Named Entity Recognition (NER) and part-of-speech tagging on the text. Specifically, we tag all words and filter the results to retain only those identified as nouns for the final keyword set. To accomplish the NER and tagging tasks, we utilize the Natasha library, which provides robust tools for Russian language processing. This approach ensures that the extracted keywords are semantically meaningful and relevant, improving the quality and interpretability of the keyword extraction process.

1. **Mechanism**: Each original review text (e.g., in Russian) is first translated into an intermediate language (e.g., English) using a neural machine translation (NMT) model (e.g., `Helsinki-NLP/opus-mt-ru-en`).

2. Subsequently, the translated text is translated back into the original language (e.g., Russian) using another NMT model (e.g., `Helsinki-NLP/opus-mt-en-ru`).

3. This process typically yields a new text that is semantically similar to the original but often exhibits variations in wording, phrasing, and sentence structure.

The generated augmented texts ($text_{aug}$) are then paired with their original counterparts ($text_{orig}$) to form positive pairs for the instance-level contrastive loss. To handle cases where augmentation might produce nonsensical or drastically different text, a filtering step is applied: if the absolute difference in character length between an original text and its augmentation exceeds a predefined threshold (e.g., 40 characters), the augmented text is replaced with a copy of the original text. This ensures that the augmentations remain reasonably faithful to the source content. For our experiments, these augmentations are pre-generated offline and stored alongside the original data to expedite the training process.

5 Evaluation Metrics

To assess the performance of our approach, we employ a set of diverse evaluation metrics, catering to both the quality of learned representations from the self-supervised phase and the final classification performance from the supervised phase.

Metrics for Supervised Classification Performance

These metrics are primarily used during and after Phase 2, where the model performs direct review classification.

- **Accuracy (Acc)**: This is the proportion of reviews that are correctly classified by the model. It is calculated as:

  $$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}} \tag{3}$$

  While intuitive, accuracy can be misleading on imbalanced datasets.

- **F1-Score (F1)**: The F1-score is the harmonic mean of precision and recall, providing a more balanced measure, especially when class distribution is uneven. It is calculated for each class and can be averaged (e.g., macro, micro, weighted) across classes. For a binary case or per-class:

  $$F1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4}$$

  where Precision = TP / (TP + FP) and Recall = TP / (TP + FN). We typically report the weighted F1-score to account for class imbalance.

Metrics for Unsupervised Representation Quality / Clustering

These metrics are particularly relevant for evaluating the quality of embeddings learned during the self-supervised Phase 1, often by applying a clustering algorithm (e.g., K-Means) to the learned features and then evaluating the resulting clusters.

- **Silhouette Coefficient (Silhouette Score)**: **Justification**: The Silhouette Coefficient measures how well-separated clusters are and how compact individual clusters are. It provides a measure of the appropriateness of the clustering structure found in the learned embedding space. A higher Silhouette Score indicates that the embeddings form more distinct and dense clusters. **Formula**: For a single sample i, let $a(i)$ be the mean distance between i and all other points in the same cluster, and let $b(i)$ be the mean distance between i and all points in the nearest neighboring cluster. The silhouette coefficient $s(i)$ for sample i is:

  $$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}} \tag{5}$$

  The overall Silhouette Score is the mean of $s(i)$ over all samples. Values range from -1 to $+1$, where $+1$ is optimal.

- **Normalized Mutual Information (NMI)**: **Justification**: If ground-truth class labels are available for the validation set (even if not used during the self-supervised training), NMI can quantify the agreement between the clustering result (obtained by applying K-Means to the learned embeddings) and these true labels. It measures the mutual dependence between the two sets of assignments, normalized to account for the number of clusters/classes. A higher NMI indicates that the learned representations group instances in a way that aligns well with the true underlying class structure. **Formula**: Given two sets of assignments (true labels U and predicted cluster labels V), NMI is defined as:

  $$\text{NMI}(U, V) = \frac{I(U, V)}{\sqrt{H(U)H(V)}} \tag{6}$$

  where $I(U, V)$ is the mutual information between U and V, and $H(U)$ and $H(V)$ are the entropies of U and V, respectively. NMI values range from 0 (no mutual information) to 1 (perfect correlation).

6 Results

Our method achieved an accuracy of 0.61, the normalized mutual information score was 0.35, the F1-score was 0.60.

7 Conclusion

This paper presented an application for categorizing restaurant reviews into key food-related categories and assigning relevance scores to each. We deeply engaged with the task of text data clustering, exploring various algorithms and methods applicable to review analysis. It was intersting task.

"В ресторане подают потрясающую рыбу — свежую и идеально приготовленную.
Особенно понравился лосось с лимонным соусом. Мясо
тоже было на высоте, сочное и нежное, особенно стейк средней прожарки.",
"Люблю мясо, и здесь его готовят просто великолепно. Курица была сочной
и ароматной, а говядина — таяла во рту. Рыба тоже достойна похвалы,
особенно блюда из морского окуня.",
"Рыба в этом ресторане просто фантастическая — свежая, с хрустящей
корочкой и нежным вкусом. Мясо, к сожалению, показалось немного
пересушенным, но в целом качество хорошее.",
"Обожаю блюда из мяса, и здесь я не разочаровался. Шашлык был отлично
замаринован и приготовлен. Рыба тоже присутствовала в меню и была
свежей и вкусной, особенно дорадо на гриле.",
"Рыба и мясо — это две главные причины, почему я возвращаюсь в этот
ресторан. Рыба всегда свежая, а мясные блюда — сочные и насыщенные
вкусом. Особенно рекомендую попробовать баранину и сибаса.",
"Вкус рыбы здесь поражает — она всегда свежая и с отличным гарниром.
Мясо также на высоте, особенно стейки из говядины и свинины.
Обслуживание дружелюбное, атмосфера уютная.",
"Рыба и мясо — главные звезды меню. Рыба приготовлена с душой, а мясо —
просто тает во рту. Особенно понравились блюда из лосося и говяжьи
ребрышки на гриле.",
"Рыба была свежей и нежной, приготовлена идеально. Мясо — сочное и
ароматное, особенно понравился стейк из мраморной говядины. Атмосфера
ресторана располагает к приятному отдыху.",
"Люблю мясо, и здесь его готовят очень вкусно. Рыба тоже на уровне,
особенно блюда из трески и форели. Порции большие, а качество продуктов
отличное.",
"Рыба и мясо — лучшие блюда в этом ресторане. Рыба всегда свежая и
вкусная, а мясо — сочное и ароматное. Особенно рекомендую попробовать
лосось и ребрышки на гриле."

Table 1: Input samples.

"keywords":["name":"рыба","score":4.6,
"name":"мясо","score":4.7,
"name":"атмосфера","score":5,
"name":"меню","score":4,
"name":"качество","score":4.5]

Table 2: Output

References

[1] Daniel Miptstudent, "NLP with Contrastive Loss for Keyword Extraction," Kaggle, 2023. [Online]. Available: `https://www.kaggle.com/code/danielmiptstudent/nlp-con-loss`

[2] E. Campos, L. Mangaravite, A. Pasquali, et al., "YAKE! Keyword Extraction from Single Documents using Multiple Local Features," arXiv preprint arXiv:2212.14366, 2022. [Online]. Available: `https://arxiv.org/pdf/2212.14366v1`

[3] A. Papagiannopoulou and G. Tsoumakas, "A Review of Keyphrase Extraction," arXiv preprint arXiv:2203.12000, 2022. [Online]. Available: `https://arxiv.org/pdf/2203.12000v2`