

Prédiction d'emojis par embedding de posts sur des réseaux sociaux

Ryan Belaib¹, Antoine Bretzner¹

Résumé

Les émojis occupent une place centrale dans la communication sur les réseaux sociaux, où ils enrichissent les messages textuels en transmettant des émotions et des nuances contextuelles. Leur intégration dans les modèles de traitement du langage reste un défi en raison de leur usage évolutif et de leur polysémie. De nombreux systèmes existants permettent de recommander des émojis à l'utilisateur, mais ces derniers ne prennent pas forcément en compte le contexte actuel d'utilisation d'un émoji. Par exemple, l'émoji 🤔, dont la description Unicode est `loudly-crying-face` est plutôt utilisé pour exprimer de la joie ou de l'euphorie, à la manière de l'émoji 😄. On cherche donc à regrouper sémantiquement les émojis dans leur contexte actuel d'utilisation. Dans ce travail, nous développons un système de prédiction d'emojis basé sur des embeddings issus de publications sociales, capturant à la fois leur contenu émotionnel (qui sera en accord avec leur sémantique actuelle) et leur contexte d'utilisation. Notre méthode utilise des embeddings contextuels issus du modèle Word2vec [5] et une forêt aléatoire [1] pour classer les posts. Nous projetons finalement les embeddings moyens pour chaque émoji dans un espace de dimension 2 en utilisant t-SNE [7] afin de visualiser les proximités sémantiques existant entre-eux. Le code de l'étude est disponible sous ce lien² et les données et modèles sont disponibles sur ce dépôt Seafile³. Antoine Bretzner a contribué à la création du jeu de données et à la rédaction du rapport. Ryan Belaib a contribué à la partie algorithmique de l'étude et à la rédaction du rapport.

-
1. Télécom Physique Strasbourg, Université de Strasbourg
E-mail: {ryan.belaib, antoine.bretzner}@etu.unistra.fr
 2. <https://github.com/Antoine-BRETZNER/Emoji-Prediction>
 3. <https://seafile.unistra.fr/d/fdc6fad717c24ec0a6c7/>

Mots-clés : émoji, embedding, Word2vec, système de recommandation, t-SNE, traitement automatique du langage naturel

1. Introduction

Les émojis sont devenus incontournables lors de nos communications. Nous les utilisons pour exprimer nos émotions et ils peuvent même être associés à des entités ou des concepts. Il est donc important de les prendre en considération lors de travaux de traitement du langage naturel ou d'analyse de sentiments par exemple. De nos jours, les utilisateurs doivent consulter un large catalogue d'émojis pour faire leur choix et d'autres encore sont ajoutés aux catalogues avec le temps : il apparaît donc comme convenable de pouvoir proposer aux utilisateurs des émojis appropriés et les plus proches possible de la sémantique portée par leurs messages. La plupart des modèles de traitement du langage naturel "ignorent" les émojis et les considèrent comme étant des métadonnées bien qu'ils soient chargés de sémantique. La plupart des systèmes de recommandation suggèrent un émoji en fonction de la description Unicode de ce dernier, sans considérer le sens du message. De nos jours, la recherche sur le sujet s'est bien établie et quelques systèmes de recommandation ont vu le jour. Ils ont obtenu de bonnes performances pour la plupart comme [8] avec 65% de précision. Cependant, ces systèmes se concentrent sur un nombre limité d'émojis et ne capturent pas l'évolution de leur sémantique de nos jours pour les plus anciens. De plus, la précision n'apparaît pas comme étant une métrique appropriée au problème : à un post peuvent correspondre plusieurs émojis, ce qui ne contribue pas toujours favorablement à la précision. Nous préférons donc évaluer les performances de notre modèle sur la sémantique de l'émoji qu'il prédira. La majorité des travaux sur le sujet ont été réalisés sur des posts en anglais ou en chinois [4] et pas en français : la sémantique portée par les émojis change aussi en fonction de la localisation de l'utilisateur. Dans notre étude, nous créerons des embeddings d'émojis en utilisant les embeddings des posts associés à chaque émoji afin de pouvoir réaliser une prédiction sur de nouveaux posts et nous visualiserons les proximités sémantiques entre les émojis. Pour cela, nous constituerons un jeu de données de posts avec leurs émojis associés à l'aide de /textitscraping sur des plateformes comme X et Reddit. Nous expliciterons aussi, dans un premier temps, la notion d'embedding et le fonctionnement du modèle Word2vec.

2. Embeddings et Word2vec

2.1 L’embedding : une représentation vectorielle d’un mot

La représentation vectorielle d’un mot, communément appelée *embedding*, est une technique de traitement automatique des langues (*Natural Language Processing*, NLP) qui consiste à représenter les mots par des vecteurs dans un espace continu de faible dimension. Contrairement aux représentations traditionnelles basées sur des vecteurs clairsemés (comme le *one-hot encoding*), les embeddings capturent les similarités sémantiques entre les mots en plaçant des mots sémantiquement proches à des distances réduites dans l’espace vectoriel.

Un embedding repose sur l’idée que le contexte dans lequel un mot apparaît porte des informations sur son sens. Cette approche est résumée par le principe de distributionnalité : les mots ayant des significations similaires apparaissent dans des contextes similaires [2]. Les modèles d’*embedding* les plus populaires apprennent ces représentations vectorielles à partir de grands corpus textuels en maximisant la probabilité des cooccurrences entre les mots et leurs contextes.

Par exemple, pour une phrase comme : **le chat dort sur le canapé**, le mot **chat** est souvent associé à des mots comme **animal** ou **félin** ; la vecteur de cette phrase pourrait donc être proche de celui des phrases contenant ces derniers mots.

Considérons un espace vectoriel à trois dimensions pour simplifier l’exemple. Supposons que les vecteurs d’embedding suivants aient été appris pour trois mots :

$$\text{chat} = [0.8, 0.5, 0.1], \quad \text{chien} = [0.9, 0.4, 0.2], \quad \text{voiture} = [0.1, 0.2, 0.9].$$

Les vecteurs **chat** et **chien** sont proches dans cet espace, reflétant leur similarité sémantique, tandis que **voiture** est éloigné, car son sens est différent.

Une mesure couramment utilisée pour quantifier cette similarité est le *cosinus de l’angle* entre les vecteurs. Par exemple, le cosinus entre **chat** et **chien** est donné par :

$$\cos(\theta) = \frac{\vec{v}_{\text{chat}} \cdot \vec{v}_{\text{chien}}}{\|\vec{v}_{\text{chat}}\| \|\vec{v}_{\text{chien}}\|}$$

Les embeddings sont utilisés dans de nombreuses applications du NLP, comme l’analyse de sentiments ou la traduction automatique par exemple. Ils jouent aussi un rôle essentiel

dans la réduction du "fléau de la dimension", un problème récurrent en intelligence artificielle. Sans ces représentations vectorielles, les mots sont encodés sous forme de vecteurs one-hot, ce qui produit des données éparpillées où chaque mot est un point isolé dans un espace de très haute dimension, difficile à exploiter. Les embeddings transforment cet espace fragmenté en un espace dense et compact, permettant de capturer des relations sémantiques et syntaxiques.

Par exemple, dans un espace vectoriel d'embedding, une relation telle que **roi** - **homme** + **femme** = **reine** devient explicite et vérifiée grâce à la structure géométrique sous-jacente.

2.2 Word2vec

Word2vec est une technique introduite par Mikolov et al. en 2013 [5] pour apprendre des représentations vectorielles denses et continues des mots, en exploitant leur contexte d'apparition dans un corpus. Deux architectures principales sont utilisées : Continuous Bag of Words (CBOW) et Skip-Gram.

Dans le modèle CBOW, l'objectif est de prédire un mot cible donné à partir de ses mots contextuels, tandis que le modèle Skip-Gram inverse cette relation en apprenant à prédire les mots contextuels à partir d'un mot cible. Mathématiquement, Word2vec cherche à maximiser une fonction de probabilité conditionnelle basée sur les cooccurrences des mots dans une fenêtre de contexte de taille k . Soit w un mot cible et $\mathcal{C}(w)$ l'ensemble des mots contextuels dans cette fenêtre ; l'objectif est de maximiser la somme des probabilités logarithmiques suivantes (donc une log-vraisemblance) :

$$\mathcal{L} = \sum_{w \in \text{Corpus}} \sum_{c \in \mathcal{C}(w)} \log P(c|w),$$

où $P(c|w)$ représente la probabilité qu'un mot contextuel c soit observé autour du mot cible w . Cette probabilité est modélisée à l'aide de la fonction softmax :

$$P(c|w) = \frac{\exp(\mathbf{v}_c \cdot \mathbf{v}_w)}{\sum_{c' \in \mathcal{V}} \exp(\mathbf{v}_{c'} \cdot \mathbf{v}_w)},$$

où \mathbf{v}_w et \mathbf{v}_c sont les représentations vectorielles respectives des mots w et c , et \mathcal{V} est le vocabulaire total. Toutefois, le calcul de cette fonction softmax est coûteux, car il nécessite de normaliser sur tous les mots du vocabulaire. Pour contourner cette limitation, des techniques d'approximation comme l'échantillonnage négatif ou le softmax hiérarchique sont utilisées.

L'échantillonnage négatif simplifie le problème en reformulant l'objectif comme une tâche de classification binaire, où l'on cherche à distinguer les paires (w, c) observées dans le

corpus de paires dites négatives générées artificiellement. La quantité à maximiser devient alors :

$$\log \sigma(\mathbf{v}_c \cdot \mathbf{v}_w) + \sum_{i=1}^n \mathbb{E}_{c' \sim P_n} [\log \sigma(-\mathbf{v}_{c'} \cdot \mathbf{v}_w)],$$

où $\sigma(x) = \frac{1}{1+\exp(-x)}$ est la fonction sigmoïde, P_n est une distribution de bruit utilisée pour échantillonner les mots négatifs, et n est le nombre de mots négatifs échantillonnés pour chaque paire cible-contexte.

Le softmax hiérarchique sert à accélérer le calcul des probabilités $P(c|w)$, qui nécessite normalement une normalisation sur l'ensemble du vocabulaire \mathcal{V} , souvent très grand. Cette méthode repose sur une décomposition arborescente du vocabulaire, où chaque mot est une feuille de l'arbre, et chaque chemin de la racine à une feuille représente des décisions binaires. La probabilité d'un mot cible c donné un mot w est alors décomposée comme le produit des probabilités des décisions prises le long du chemin correspondant dans l'arbre:

$$P(c|w) = \prod_{j=1}^{L(c)} P(d_j|w),$$

où $L(c)$ est la longueur du chemin menant au mot c , et d_j est une décision binaire prise à chaque nœud (par exemple, aller à gauche ou à droite). Les probabilités binaires $P(d_j|w)$ sont modélisées à l'aide de la fonction sigmoïde $\sigma(x) = \frac{1}{1+\exp(-x)}$, et chaque nœud de l'arbre est associé à un vecteur.

Cette approche réduit considérablement le coût computationnel de l'évaluation et de la mise à jour des probabilités, passant de $O(|\mathcal{V}|)$ pour un softmax classique à $O(\log_2(|\mathcal{V}|))$ dans un arbre bien équilibré. Par exemple, pour un vocabulaire de 1 million de mots, le calcul passe d'un million de termes à environ 20, rendant l'entraînement sur de grands corpus beaucoup plus rapide. Le choix de la structure de l'arbre peut également avoir un impact sur les performances ; des arbres équilibrés ou optimisés en fonction de la fréquence des mots sont souvent utilisés.

Word2vec est paramétré par la dimension d des vecteurs, la taille de la fenêtre k , le taux d'apprentissage α , et le nombre de mots négatifs n pour l'échantillonnage négatif. Ces hyperparamètres influencent significativement la qualité des embeddings et doivent être ajustés en fonction de la tâche et du corpus.

3. Création du jeu de données

Notre jeu de données est constitué de 5150 posts en français, *scrapés* depuis des plateformes comme X (anciennement Twitter), Reddit ou même générés grâce à de l’IA générative pour des raisons de praticité. À chaque post est associé un emoji qui est apparu dans ce dernier. En raison des coûts d’accès à l’API de la plateforme X, nous avons utilisé la librairie **BeautifulSoup** en Python, ce qui rendait la récupération des posts de cette plateforme bien plus ardue. En revanche, L’API Reddit a pu être utilisée.

En proportion, notre jeu de données contient 60% de posts issus de Reddit, 30% de posts issus de X et 10% de posts générés par IA pour l’enrichir.

Voici quelques statistiques sur notre jeu de données :










Statistique	Valeur
Émojis les plus représentés	
	172
	154
	148
	134
	96
Longueur moyenne des phrases	18.66 mots
Nombre d’emojis distincts	592

Table 1: Statistiques sur les phrases et emojis

Parfois, plusieurs émojis peuvent apparaître dans un post : nous avons donc dû ”dupliquer” certains posts et leur associer plusieurs émojis (par exemple, *bonne année 2025 !*   sera transformé en *bonne année 2025 !*  et *bonne année 2025 !* ). L’API Reddit générerait des doublons des posts lors de son utilisation, ces doublons ont été supprimés avant traitement final pour ne pas biaiser les sémantiques de certaines phrases. Nous avons aussi supprimé tous les liens hypertextes et mentions pouvant apparaître dans les posts car ils contiennent une sémantique négligeable par rapport aux mots et ils seront mal traités par Word2vec, qui a été pré-entraîné sur un corpus de mots usuels.

4. Protocole expérimental et choix d'implémentation

Afin de pouvoir générer des embeddings représentatifs d'un tweet, il a fallu *tokeniser* ces derniers, donc les transformer en listes de mots (tokens) pour former le corpus à passer à Word2vec. Pour cela, on utilise le tokenizer de la librairie NLTK en Python.

On entraîne par la suite Word2vec, dans son implémentation **gensim**, sur le corpus de mots. Nous avons choisi de l'utiliser dans son architecture Skip-Gram qui est plus appropriée que CBOW pour notre objectif : elle est plus flexible pour capturer des relations complexes entre des mots peu fréquents ou très spécialisés car on se concentre plus sur la prédiction du contexte entourant ces mots.

Nous avons expérimenté plusieurs tailles de fenêtre de contexte pour le modèle, et les résultats les plus proches de la sémantique des posts ont été donnés pour une taille de 3 mots, ce qui paraît compréhensible au vu de la structure d'un post : ils sont souvent courts et utilisent des mots très chargés de sémantique pour exprimer une émotion précise "rapidement".

Pour spécialiser notre modèle, nous avons aussi choisi de considérer tous les mots du corpus d'entraînement. En fixant un seuil minimum de fréquence (5 par exemple), les performances du modèle étaient du même ordre. Le modèle génère des vecteurs de dimension 300.

À cette étape, le modèle Word2vec a donc généré des vecteurs pour chaque mot du corpus d'entraînement. Pour modéliser vectoriellement un post, nous effectuons la moyenne des vecteurs des mots le constituant. Ainsi, chaque post est localisé dans l'espace et deux posts proches sémantiquement le seront au sens de la distance associée à cet espace.

Afin de classer les posts, on utilise une forêt aléatoire avec 200 estimateurs. Comme la métrique de précision ne nous intéresse pas directement dans notre étude, utiliser une validation croisée pour trouver les meilleurs paramètres nous semblait être inadéquat. Nous avons cependant trouvé que le nombre d'estimateurs influait de manière minime sur la qualité sémantique des émojis prédits, et nous avons donc choisi ce nombre en conséquence. Nous avons aussi essayé de nombreux autres classificateurs : SVM, perceptron multi-couches, k plus-proches voisins... Les forêts aléatoires étaient les plus adaptées à notre étude et produisaient les meilleurs résultats.

Afin de visualiser les potentiels groupes d'émojis formés par les embeddings, on calcule le vecteur moyen de chaque classe (donc de chaque emoji) et on projette ces vecteurs dans un espace de dimension 2 en utilisant t-SNE dans son implémentation **sklearn**. Une perplexité de 5 permet de bien distinguer les différents groupes tout en gardant une

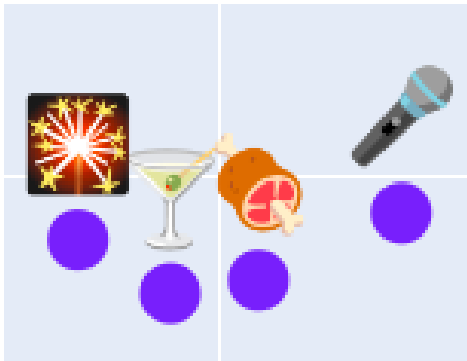
certaine proximité entre-eux s'ils sont sémantiquement proches.

5. Conclusion

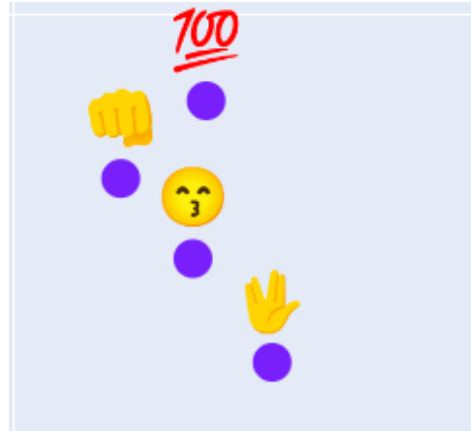
Tweet	Emoji réel	Emoji prédit
le Toit de l'Indochine INRATABLE	🇫🇷	🤖
Bon petit mood tristou de la mi-semaine !	😭	❤️
À travers mon entreprise, c'est la France qui rayonne à l'international et qui augmente son attractivité pour les futurs investisseurs étrangers.	🇫🇷	📈
Recette de pain maison	🍞	😂
Dernier jour de tournage	👋	🎬

Table 2: Comparaison entre les emojis réels et prédits.

En projetant les vecteurs pour chaque émoji, on ne parvient pas à distinguer de groupe sémantique clair pour nos émojis, présents en trop grand nombre dans notre jeu de données et dans une répartition inégale. On remarque cependant, en regardant en détail, la formation de petits groupes pouvant être interprétés :



(a) La fête (nouvel an) ?



(b) Émojis souvent utilisés ensemble

Le problème majeur du système vient du jeu de données : il contient trop peu de posts, de longueur trop variées, avec trop d'émojis distincts et en proportions inégales. Nous avons

donc essayer d'équilibrer le jeu de données en ne gardant que les émojis n'étant présents qu'au dessus d'un certain seuil ; les prédictions sont donc bien meilleures pour les tweets associés sémantiquement à ces émojis, mais bien pires pour les tweets plus "spécialisés". Il-y-a donc un compromis nombre d'émojis-proportion-nombre de posts à satisfaire lors de la construction du jeu de données : les études conduites sur le sujet le montrent bien, comme [3] qui n'utilise que les émojis simples (😄, 😊...) dans environ 700 000 posts.

Il nous paraît cependant être important de capturer un grand nombre d'émojis dans le contexte actuel, afin de répondre à la diversification sémantique des posts de nos jours et à l'expansion continue du catalogue d'émojis mis à disposition des utilisateurs. Pour des raisons matérielles, nous n'avons pas pu récupérer beaucoup de posts différents en français afin de constituer un jeu de données adapté à notre étude.

Cependant, nous avons récupéré l'un des rares jeux de données⁴ possédant la même structure que le nôtre en anglais. Ce jeu se concentre uniquement sur 20 émojis et contient 69832 posts issus de X. Voici quelques exemples de prédictions :

Tweet	Emoji réel	Emoji prédit
Be still my heart	❤️	❤️
Real tears	😭	😭
Good morning LA!	😎	☀️
Life is good ! Our broccoli is showing up, also collards, cauliflower, Brussels sprouts,...	☀️	😂

Table 3: Comparaison entre les emojis réels et prédits (anglais).

Les prédictions sont donc beaucoup plus appropriées avec beaucoup de posts et un nombre limité d'émojis. On remarque ici que la manque d'émojis utilisés est regrettable dans le cas du dernier post, qui, dans l'hypothèse d'un jeu de données idéal, serait vraisemblablement associé à des émojis comme 🥦 ou 🥬.

La projection avec t-SNE sur ce jeu de données donne le résultat suivant :

4. <https://www.kaggle.com/datasets/hariharasudhanas/twitter-emoji-prediction>

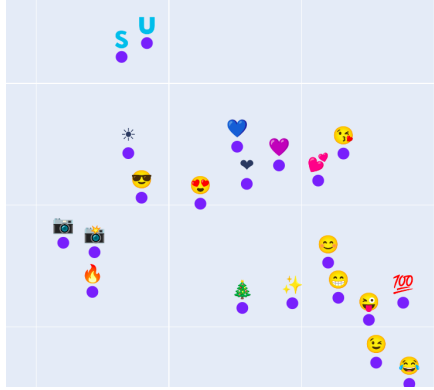


Figure 2: Projection des embeddings moyens (anglais).

Les groupes sémantiques sont clairs dans ce cas (coeurs, joie/humour, soleil, caméras) mais peu nombreux en raison du manque de diversité en termes d'émojis du jeu de données choisi.

6. Conclusion et ouverture

Au cours de cette étude, nous avons exploré la possibilité de prédire des émojis pertinents pour des posts sur des réseaux sociaux écrits en français, en nous appuyant sur des techniques d'apprentissage automatique et de traitement du langage naturel. Le contexte et l'importance des émojis dans les communications numériques, soulignant leur richesse sémantique et les défis qu'ils posent pour des systèmes de recommandation, constituent les motivations de notre système. Afin d'aborder ce problème, nous avons utilisé des embeddings vectoriels comme représentation sous-jacente, en nous appuyant sur le modèle Word2vec dans son architecture Skip-Gram, particulièrement adapté pour capturer des relations complexes entre mots contextuels.

La création de notre jeu de données a constitué une étape clé, bien qu'elle ait été limitée par les ressources disponibles. Nous avons collecté et nettoyé environ 5 150 posts en français provenant principalement de Reddit et de X, en y associant les émojis présents dans les messages. Cette base a permis d'entraîner notre modèle, qui produit des vecteurs représentatifs pour chaque message. Ces vecteurs ont ensuite été utilisés pour classifier les posts, avec une forêt aléatoire comme classificateur.

Les résultats obtenus montrent que, bien que notre modèle parvienne à prédire des émo-

jis pertinents dans certains cas, il présente des limites notables : la faible quantité de données, l'inégalité dans la répartition des émojis, et la diversité des longueurs des messages ont réduit la qualité générale des prédictions. la projection des embeddings moyens dans un espace de dimension réduite n'a pas permis de distinguer clairement des groupes sémantiques, sauf pour des émojis fréquemment utilisés.

Ces limites mettent en évidence le besoin d'un jeu de données plus vaste et mieux équilibré, afin de capturer à la fois la diversité des émojis et la richesse sémantique des messages. Par comparaison, les tests effectués sur un jeu de données en anglais, bien plus volumineux mais moins varié en termes d'émojis, ont démontré des performances significativement supérieures, avec des groupes sémantiques plus cohérents. Ce constat souligne l'importance cruciale de la qualité et de la quantité des données dans ce type de système.

Pour aller plus loin, d'autres modèles d'embeddings auraient pu être explorés. Par exemple, l'utilisation de modèles pré-entraînés comme BERT ou FastText aurait permis de capturer des nuances contextuelles plus fines grâce à leur capacité à prendre en compte le contexte global d'un message. Une étude récente sur BERT [6] a montré que ce modèle excelle dans la prédiction d'émojis, avec une précision de 75%.

Il est aussi essentiel de considérer les implications éthiques de tels systèmes. La recommandation d'émojis, bien que ludique en surface, peut influencer la manière dont les utilisateurs expriment leurs émotions ou leurs idées. Ces systèmes peuvent également refléter, voire amplifier, des biais présents dans les données d'entraînement, comme des stéréotypes culturels ou des préjugés liés à certaines communautés ou groupes linguistiques. Une attention particulière doit donc être portée à la diversité des données utilisées et à la transparence des algorithmes mis en œuvre.

Cette étude met en lumière les défis et les opportunités liés à la construction de systèmes de recommandation d'émojis. Si les résultats obtenus sont encourageants, ils appellent à des recherches plus approfondies, combinant des approches de pointe en traitement automatique du langage naturel et une réflexion rigoureuse sur les questions éthiques et sociétales qu'ils sont amenés à soulever.

References

- [1] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.
- [2] J. R. Firth. A synopsis of linguistic theory 1930–1955. In *Studies in Linguistic Analysis*, pages 1–32. Blackwell, Oxford, 1957. Reprinted in F. R. Palmer (Ed.), *Selected Papers of J. R. Firth 1952–1959*, London: Longman, 1968.

- [3] Gaël Guibon and Patrice Bellot. From emoji usage to categorical emoji prediction. In *Lecture Notes in Computer Science*. February 2023. Citations: 11, Reads: 1,745.
- [4] Chuchu Liu, Fan Fang, Xu Lin, Tie Cai, Xu Tan, Jianguo Liu, and Xin Lu. Improving sentiment analysis accuracy with emoji embedding. *Journal of Safety Science and Resilience*, 2:246–252, 2021.
- [5] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. 2013. Last revised on 7 Sep 2013.
- [6] Muhammad Osama Nusrat, Zeeshan Habib, Mehren Alam, and Saad Ahmed Jamal. Emoji prediction in tweets using BERT. *arXiv preprint arXiv:2307.02054*, 2023.
- [7] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, November 2008.
- [8] Ruibin Xie, Zhiyuan Liu, Rui Yan, and Maosong Sun. Neural emoji recommendation in dialogue systems. *arXiv preprint arXiv:1612.04609*, 2016.