Dependable AI Assignment-1 Report

Integrated Gradients

Aditya Rathor

B22AI044

Vishesh Sachdeva

B22AI050

Table of Contents

Pipeline for Integrated Gradients

1. **Feature Extraction**: 2 methods are used for finding embeddings(features) for the description of train and test images:
    a. **Glove**:
    - As glove computes embedding at word level so before computing embedding directly we pass the sentence through the text preprocessing pipeline which involves:
        - Lowercasing, stopword removal using nltk's stopwords
        - Spacy to remove punctuation, number and special character
    - After this using the 'glove.6B.50d.txt' a 50d vector is generated for each word and to get embedding for the sentence we do average pooling

    b. **SBert**:
    - SentenceTransformer model (all-MiniLM-L6-v2) is used to convert the input text into a 384d vector
    - Sentence embedding models capture contextual and semantic meaning of the entire and doesn't lose context and word order.

2. **Training AnnexML**: AnnexML is trained on these extracted features

3. **Finding the intermediate samples:** For selected samples of IG we find samples between the baseline to original feature by following the 2 methods
    a. Scaling Feature Value between Baseline to Original Input: Progressively scaling the features from 0 to their original value in num_steps increments.
    b. Incrementing Individual Features: After scaling, each feature is individually incremented by adding a small step_size value, which helps approximate the impact of each feature by creating slight variation for it

    Further for each sample we compute scores for labels of using the annexML predict function

4. **Approximation For IG:** As for AnnexML we can't find the gradient so we approximate by measuring how class predictions change as input features are gradually scaled.

    The difference in predicted label scores between consecutive scaled samples estimates how much a feature influences the model's output for that label. A feature with a significant score increase for a small increment has higher importance.

5. **Visualising the Obtained Attribution:** Since the attribution scores are computed for each feature embedding while the input is raw text, these scores may not be directly meaningful. Therefore, based on the method used to compute embeddings, we derive attribution scores for each word as follows:

    a. **Glove**: As embedding in this are computed based on word level so what we do is we again compute embedding of preprocessed sentence's words and take a dot product between these word embedding and attribution scores divided by the norm of attribution scores to estimate the importance for each word. Thereby ensuring the projection feature-level scores back onto individual words.

    b. **SBert**: Backpropagate gradients from the sentence embedding to word embeddings by distributing them proportionally to how each word contributes to the sentence embedding, typically using the Jacobian of the embedding transformation.

Implementation

1. **Extracting the labels, raw texts and scaled features:**
   - get_test_data(lines_to_take) – Uses a list of indices provided as an argument to extract file paths from iaprtc_test_list.txt, which contains paths to annotation files. Since these annotation files contain only descriptions, the function loads test labels from the IAPRTC_12_TestLabels.mat file, converts one-hot encodings to label indices, and retrieves text descriptions from the annotation files.
   - generate_all_scales(test_path, scaled_test_path, step_size, num_steps): generates scaled versions of feature vectors from a given test dataset. It reads feature values from test_path, progressively scales them from a baseline (0) to their original values over num_steps, and perturbs individual features by adding step_size.

2. **Generating Word-Level Embeddings:**
   Defined the Glove class that contains the following methods:
   - load_glove_model() – Loads the GloVe embeddings from a file and stores them in a dictionary for quick lookup.
   - get_glove_embeddings(words) – Retrieves GloVe embeddings for a list of words, returning zero vectors for missing words.
   - tokenize_sentence(sentence) – Tokenizes a sentence using spaCy while filtering out non-alphabetic tokens.
   - sentence_to_glove_embedding(sentence) – Tokenizes, removes stopwords, retrieves GloVe embeddings for each word, and returns both embeddings and tokens.
   - create_testfile_w_embedding(text_path, test_path, labels, texts) – Prepares a test file with sentence embeddings and labels for model inference.After average pooling the labels along with features are saved in the test_path.

   For SBert:

   - create_testfile_w_embedding(self, text_path, test_path, labels, texts) – Converts sentences into SBERT embeddings using "embedding=tf_model.encode(description)" and stores them with corresponding labels for model inference.
   - backpropogate(self, sentence, gradients) – Computes token-level attribution scores by enabling gradients on SBERT embeddings given gradients with respect to 384d output vector

- generate_colored_text(self, text, word_scores) – Generates HTML-based colored text to visualize word importance based on attribution scores.
- get_word_gradients(self, texts, test_pred_path, gradient_list_row, classes) – Extracts and normalizes token gradients to highlight influential words for each predicted class.

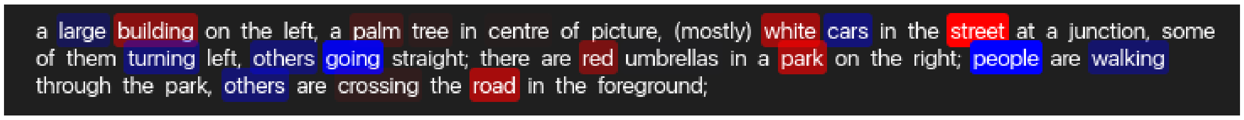
3. **Computing Integrated Gradients (IG) for Feature Importance:**
   - IG(test_path, test_pred_path, scaled_pred_path, step_size, num_steps, classes_labels) – Approximates Integrated Gradients by computing changes in class predictions across scaled feature inputs.

4. **Mapping Feature-Level Attributions to Words:**
   - Class Glove: word_level_attribution() function projects IG-based feature scores onto individual words by computing the dot product between word embeddings and attribution vectors.
   - Class SBert: backpropagate() method
     - Input IDs are tokenized, converted to embeddings via the model's word_embeddings layer, and requires_grad_() is enabled to allow gradient computation for Jacobian calculation.
     - Token embeddings are fed into the model via inputs_embeds, and a sentence embedding is generated by mean pooling the last_hidden_state.
     - Gradients with respect to the sentence embedding are back-propagated to compute the Jacobian, linking the token embeddings to the sentence embedding through partial derivatives.
     - The norm of token gradients is computed to quantify each token's contribution to the sentence embedding, enabling analysis of token-level importance.
   - Generate_colored_text(text, word_scores) – Highlights words in a sentence based on their attribution scores using color coding.

Explaining 2 predicted labels for row 1 and 3

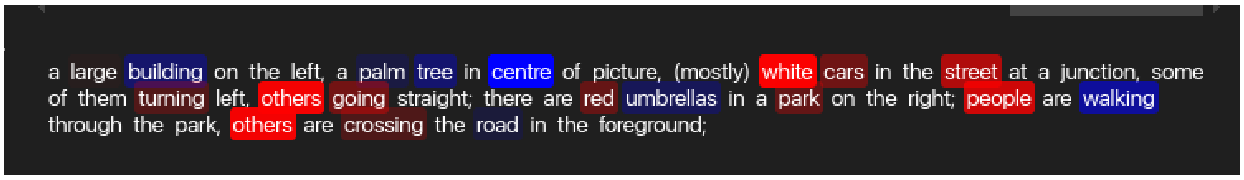
- **For Glove** :

Attribution scores for all tokens in **people** class: {'people': 0.1858, 'going': 0.1653, 'picture': 0.1138, 'mostly': 0.1081, 'cars': 0.0741, 'turning': 0.0712, 'others': 0.0671, 'walking': 0.0637, 'large': 0.0451, 'right': 0.0299, 'foreground': 0.0248, 'left': 0.0121, 'umbrellas': 0.0018, 'centre': -0.0055, 'tree': -0.0125, 'crossing': -0.0164, 'palm': -0.0337, 'junction': -0.0614, 'red': -0.0881, 'building': -0.1018, 'straight': -0.1082, 'white': -0.1233, 'road': -0.1307, 'park': -0.1316, 'street': -0.2161}

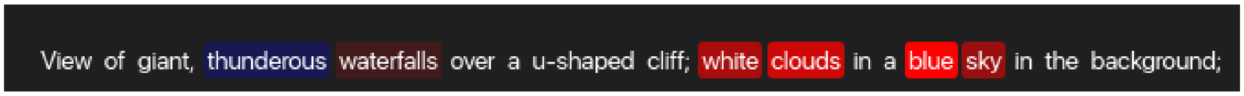

a large building on the left, a palm tree in centre of picture, (mostly) white cars in the street at a junction, some of them turning left, others going straight; there are red umbrellas in a park on the right; people are walking through the park, others are crossing the road in the foreground;

Showing Integrated Gradients for tree class

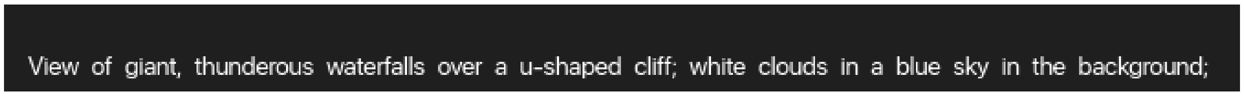
Attribution scores for all tokens in **tree** class: {'centre': 0.1059, 'left': 0.099, 'walking': 0.0587, 'tree': 0.0372, 'building': 0.0358, 'umbrellas': 0.0272, 'palm': 0.0167, 'junction': 0.0164, 'road': 0.0128, 'foreground': -0.0012, 'large': -0.0078, 'straight': -0.026, 'right': -0.0406, 'crossing': -0.0449, 'turning': -0.0476, 'park': -0.0642, 'red': -0.0662, 'cars': -0.0752, 'going': -0.0785, 'street': -0.1367, 'mostly': -0.1574, 'people': -0.1644, 'others': -0.2005, 'white': -0.2065, 'picture': -0.211}


a large building on the left, a palm tree in centre of picture, (mostly) white cars in the street at a junction, some of them turning left, others going straight; there are red umbrellas in a park on the right; people are walking through the park, others are crossing the road in the foreground;

Attribution scores for all tokens in **tree** class: {'u': 0.1686, 'shaped': 0.044, 'thunderous': 0.0389, 'background': 0.033, 'waterfalls': -0.0425, 'cliff': -0.1151, 'sky': -0.1471, 'white': -0.1652, 'giant': -0.1677, 'view': -0.2084, 'clouds': -0.2111, 'blue': -0.2701}


View of giant, thunderous waterfalls over a u-shaped cliff; white clouds in a blue sky in the background;

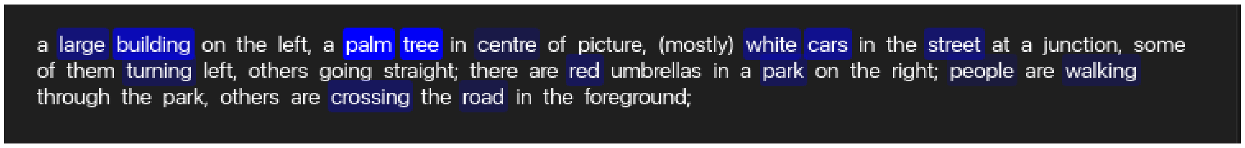
Attribution scores for all tokens in **street** class: {'view': 0.0, 'giant': 0.0, 'thunderous': 0.0, 'waterfalls': 0.0, 'u': 0.0, 'shaped': 0.0, 'cliff': 0.0, 'white': 0.0, 'clouds': 0.0, 'blue': 0.0, 'sky': 0.0, 'background': 0.0}



View of giant, thunderous waterfalls over a u-shaped cliff; white clouds in a blue sky in the background;

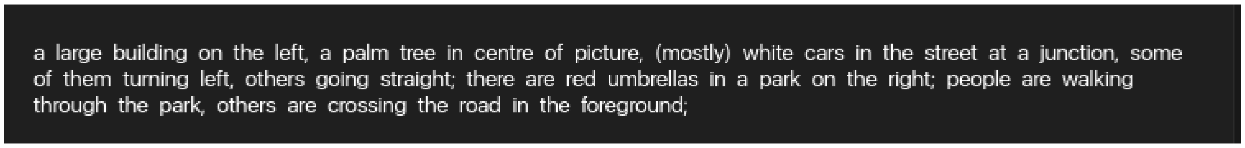
- **For SBert :**

Attribution scores for all tokens in **tree** class:[('palm', 43.92998), ('tree', 42.139717), ('umbrella', 29.895927), ('building', 29.6151), ('cars', 21.697287), ('junction', 19.27373), ('large', 17.312794), ('white', 16.862207), ('street', 14.628399), ('crossing', 11.815639), ('red', 11.060029), ('park', 10.727917), (')', 9.900093), ('turning', 9.608349), ('(', 9.411086), ('picture', 9.124996), ('mostly', 9.016131), ('##ground', 8.864766), ('centre', 7.964374), ('road', 7.7596188), ('walking', 7.0207443), ('straight', 6.410044), ('people', 5.6592946), ('left', 4.936824), ('going', 4.3976636), ('are', 4.3553987), ('there', 4.2438397), ('others', 4.2186036), ('fore', 4.1864486), ('##s', 4.1103706), ('a', 3.9534163), ('at', 3.8432486), ('through', 3.8339508), ('them', 3.7485023), (';', 3.682005), ('right', 3.6504333), (',', 3.3056965), ('some', 3.2717032), ('of', 3.0584533), ('on', 2.8458462), ('in', 2.7460718), ('the', 2.4302793)]

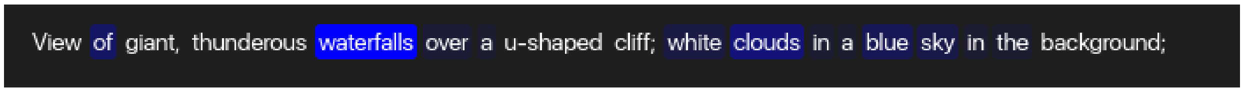


a large building on the left, a palm tree in centre of picture, (mostly) white cars in the street at a junction, some of them turning left, others going straight; there are red umbrellas in a park on the right; people are walking through the park, others are crossing the road in the foreground;

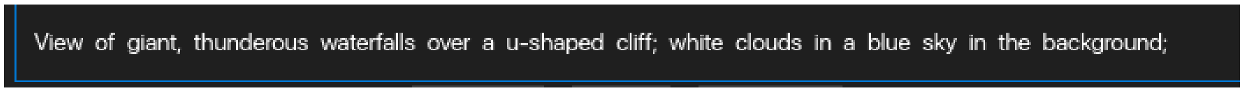
Attribution scores for all tokens in **round** class: [('umbrella', 1.0952553), ('building', 1.0863935), ('palm', 1.0740455), ('junction', 0.88571995), ('street', 0.822208), ('cars', 0.6924247), ('tree', 0.6062807), ('crossing', 0.58715504), ('large', 0.54438746), ('white', 0.5305132), ('turning', 0.41801608), ('park', 0.41088226), ('road', 0.36122653), ('red', 0.3319471), ('##ground', 0.311485), ('picture', 0.30469242), ('walking', 0.30222175), ('straight', 0.29221717), ('(', 0.27759063), (')', 0.2704045), ('mostly', 0.26546204), ('people', 0.22796716), ('centre', 0.21326546), ('right', 0.20727827), ('left', 0.2064748), ('others', 0.16672312), ('at', 0.15258299), ('through', 0.15244156), ('going', 0.1379013), ('them', 0.13762091), ('are', 0.13453563), ('a', 0.13359578), ('fore', 0.13353011), (';', 0.12762055), ('there', 0.11854774), (',', 0.10884897), ('some', 0.103949174), ('of', 0.10394025), ('##s', 0.09582077), ('on', 0.095648214), ('in', 0.09298995), ('the', 0.07707601)


a large building on the left, a palm tree in centre of picture, (mostly) white cars in the street at a junction, some of them turning left, others going straight; there are red umbrellas in a park on the right; people are walking through the park, others are crossing the road in the foreground;

Attribution scores for all tokens in **waterfall** class:[('waterfalls', 17.519295), ('cliff', 9.895741), ('clouds', 7.701156), ('thunder', 7.3913903), ('of', 6.0729823), ('giant', 5.6187735), ('sky', 5.257371), ('blue', 4.361491), ('##ous', 3.9144988), ('u', 3.5775495), (',', 3.5719674), (';', 3.5602934), ('white', 3.4113576), ('shaped', 3.3589602), ('view', 3.1778402), ('background', 2.360158), ('over', 2.2587442), ('-', 2.0048363), ('in', 1.8647801), ('a', 1.7858446), ('the', 1.722947)]


View of giant, thunderous waterfalls over a u-shaped cliff; white clouds in a blue sky in the background;

Attribution scores for all tokens in **side** class [('view', 0.0), ('of', 0.0), ('giant', 0.0), (',', 0.0), ('thunder', 0.0), ('##ous', 0.0), ('waterfalls', 0.0), ('over', 0.0), ('a', 0.0), ('u', 0.0), ('-', 0.0), ('shaped', 0.0), ('cliff', 0.0), (';', 0.0), ('white', 0.0), ('clouds', 0.0), ('in', 0.0), ('blue', 0.0), ('sky', 0.0), ('the', 0.0), ('background', 0.0)]


View of giant, thunderous waterfalls over a u-shaped cliff; white clouds in a blue sky in the background;

Observations & Analysis

- **Strong feature association for specific labels** - The attributions for the tokens if present in the sentence which is the same as the predicted labels is generally high. These words can be seen with dark blue color.

  E.g. - “people:0.1858” dominates for label “People”, “tree:42.13” dominates for label “Tree”

- **Contextual Cues and their Influence** -

For SBert We can see significant positive attribution for words (in light blue colour) that share some contextual relationship with the predicted label.

E.g. - “'cliff':9.895”, “'clouds': 7.70”, for label “Waterfall” and “palm:43.92” for “Tree”

Whereas for Glove it is not the case as word level embedding fails to capture context

- **Filtering Out Neutral or Irrelevant** - Many words have zero attribution, meaning they do not influence the prediction. These are likely common words (e.g., "the," "a",”on”) or words that do not add relevant information for the label. The model effectively filters out noise.Whereas in Glove such words are already eliminated by nltk stopwords

- **Negative Attributions Indicate Conflicting Features** - Some words get slight negative attributions(in light red color) which shows that presence of some words makes presence of predicted labels in the test image less likely.

  E.g. - “building:--0.102” for label “people” and “'mostly': -0.1574, 'people': -0.1644, 'others': -0.2005” for label “tree”

- **Weak or vague concepts** - There exists some predicted labels which do not have any significant attributions, meaning the model struggles to find defining features for it.

  E.g. - The "side" label has low attribution across all words, as that word is not present in the description, similarly with “round”

References

- Mukund Sundararajan and Ankur Taly and Qiqi Yan ***Axiomatic Attribution for Deep Networks*** - Proceedings of the 34th International Conference on Machine Learning (ICML) 2017

- https://www.youtube.com/watch?v=AHkeeyoNGp8 - Post hoc analysis of the trained network, Flaw in Integrated gradients approach due to uninteresting/flawed gradients

- Also discused with our course instructor Dr. Yashaswi Verma and course TA Rudra Dutt

- https://github.com/ankurtaly/Integrated-Gradients - Code Implementation of Integrated Gradients

- Jeffrey Pennington, Richard Socher and Christopher D.Manning. 2014 GloVe: Global Vectors for Word Representation: for Glove Embedding

- https://drive.google.com/file/d/1-sN6K_5sHzrQsAURW4XAoeknQYgkrG9R/view?usp=sharing - Model explainability using Integrated Gradients lecture slides - CSL 7370 Dependable AI course (Jan 2025) - Dr. Yashaswi Verma, IIT Jodhpur

- https://realpython.com/natural-language-processing-spacy-python/ - For filtering non-alphabetic tokens

- https://stackoverflow.com/questions/23271575/printing-bold-colored-etc-text-in-ipython-qtconsole - printing text with colored background