Lawrence Hua
LHUA
9/9/2024

**Github Link:**
https://github.com/mlip-cmu/i1-albumy-LawrenceHua/commit/0d78bc37781fb2f6507b6fedd72094da25f61ead

# Table of contents

**Technical Description** **2**
**User Interface Design Approach** **3**
**Harms** **4**
**Production Challenges** **5**

# Technical Description

In this implementation, I extended the open-source project **Albumy** to introduce two machine-learning-powered features using the Azure Computer Vision API:

1. **Automatically Generated Alternative Text for Images**
2. **Image Search by Tags**

1. The **Alternative Text Generation** feature automatically generates descriptive alt text for images when they are uploaded. This improves accessibility by providing a meaningful description for images that users might upload without providing a description. The implementation leverages the Azure Computer Vision API, specifically the `/analyze` endpoint, after adding your own endpoint/api, to process uploaded images and return captions and tags. Below are the steps:

- **Upload Route** (https://github.com/mlip-cmu/i1-albumy-LawrenceHua/blob/master/albumy/blueprints/main.py#L172-L196):
    - The `upload` route (`app/main.py`) was modified to handle image uploads. Upon saving an image, the route sends a POST request to the Azure API to generate the alternative text (alt text) and tags.
- **Azure API Integration** (https://github.com/mlip-cmu/i1-albumy-LawrenceHua/blob/master/albumy/blueprints/main.py#L40-L54):
    - The function `generate_alt_text_and_tags` was added to communicate with the Azure API, sending the image in binary format and retrieving a JSON response containing the alt text. The response is parsed, and the generated alt text is stored in the `description` field of the `Photo` model (`app/models.py`).
- **Storing Alt Text** (https://github.com/mlip-cmu/i1-albumy-LawrenceHua/blob/master/albumy/blueprints/main.py#L180-L192):
    - The alt text is stored as part of the `Photo` model in the `description` field, allowing it to be retrieved and displayed when needed. The alt text is automatically generated if the user doesn't provide one.

2. The **Image Search by Tags** feature allows users to search for images based on the objects detected in those images (tags). When an image is uploaded, tags are automatically generated via the Azure API and stored in the database. The search functionality then enables users to query the tags and retrieve relevant images. Below are the steps:

- **Tag Generation** (https://github.com/mlip-cmu/i1-albumy-LawrenceHua/blob/master/albumy/blueprints/main.py#L184-L192):
    - When the image is uploaded and analyzed by the Azure API, the detected objects (tags) are returned in the response. These tags are stored in the `tags` field of the `Photo` model, which has a many-to-many relationship with the `Tag` model. The tag data is saved to the database at the same time as the alt text.
- **Search Functionality** (https://github.com/mlip-cmu/i1-albumy-LawrenceHua/blob/master/albumy/blueprints/main.py#L92-L97):
    - The `search` route was modified to include the ability to search by tags. When a user submits a search query with the category set to "tag", the app searches for matching tags in the database and returns associated images. This is handled in the `search` route in `app/main.py`.

## User Interface Design Approach

This section outlines the interaction design choices for the two features implemented: **Alternative Text Generation** and **Image Search by Tags**. Both were designed to optimize the balance between automation and user control, ensuring a seamless user experience.

1. The **Alternative Text Generation** feature is fully automated, meaning no additional user input is required during the image upload process. When a user uploads an image, the system automatically generates a description using the Azure Computer Vision API, which is stored in the `description` field of the `Photo` model.

- **Interaction Type**: Automated.
- **Forcefulness**: Minimal – The user is not forced to provide a description, though they have the option to overwrite the generated alt text if desired.
- **Frequency**: High – Every image upload automatically triggers the alt text generation.
- **Value**: Significant – Automating alt text improves accessibility by ensuring all images have descriptions, without requiring additional effort from the user.
- **Cost**: Low – The Azure API call is efficient, and the process occurs without noticeable delays to the user.

**Justification**: The automated approach was chosen to streamline the upload process while ensuring accessibility standards are met. Given that descriptions are critical for accessibility and search engines, automating this ensures consistent coverage across all images. Additionally, allowing users to manually modify the alt text provides flexibility when needed.

**Improvement**: An improvement could involve displaying the generated alt text to the user immediately after upload, allowing them to confirm or edit it. This would enhance transparency while maintaining automation.

2. The **Image Search by Tags** feature uses a hybrid approach. Tags are automatically generated during the image upload process using the Azure Computer Vision API, but users interact with the system when they manually initiate searches based on these tags.

- **Interaction Type**: Hybrid – Tags are generated automatically, but users initiate searches based on the tags.
- **Forcefulness**: Minimal – Users are not required to engage with the tagging process unless they wish to search for specific images.
- **Frequency**: Moderate – Tag generation occurs automatically during every upload, while the search process is initiated by users on demand.
- **Value**: High – Automatically generating tags increases the discoverability of images, enhancing search capabilities for users without requiring additional effort from them.
- **Cost**: Moderate – Although tag generation is efficient, performing searches may have higher resource requirements, especially with large datasets. However, this remains within acceptable limits for most use cases.

**Justification**: The hybrid approach balances ease of use with powerful functionality. Users are not required to manage tags but can benefit from automated tagging when searching for images. This design improves the discoverability of images while maintaining a user-driven search process.

**Improvement**: Future iterations could include features such as search suggestions based on commonly used tags or allowing users to refine tags post-upload, giving them more control over how their images are categorized.

## Harms

The introduction of machine learning features for **Alternative Text Generation** and **Image Search by Tags** raises potential concerns regarding **bias**, **inaccuracy**, and **fairness**. These harms could impact user experience, accessibility, and the overall trustworthiness of the platform.

1. One potential harm is the introduction of **bias in the automatically generated descriptions**. Since machine learning models are often trained on large datasets that may contain biased representations, the alt text generated could inadvertently reflect stereotypes. For example, images depicting women in professional settings might receive less professional descriptions compared to similar images of men.

- **Impact**: Biased descriptions could reinforce harmful stereotypes, leading to alienation of certain user groups and perpetuation of societal inequalities.
- **Solution**: A bias-checking mechanism could be implemented to detect and flag potentially biased alt text, allowing manual review. Additionally, retraining the model on more diverse datasets could reduce bias. Providing users with the ability to edit alt text offers an immediate mitigation strategy.

2. Another concern is the potential for **inaccurate descriptions or tags**. The machine learning model may misinterpret images, generating incorrect or misleading descriptions. For example, an image of a sunset might be tagged as a sunrise, or an animal might be mislabeled.

- **Impact**: Inaccurate alt text and tags could degrade accessibility and reduce the effectiveness of search functionality, particularly for users who rely on accurate descriptions.
- **Solution**: Implementing a **confidence threshold** for generated alt text and tags could help filter out low-confidence results. Additionally, allowing users to edit or confirm generated descriptions and tags would reduce the risk of inaccuracy affecting the user experience.

3. Fairness issues may arise in the **Image Search by Tags** feature if the model performs better on certain types of images than others. For example, if the model is trained primarily on western objects or landscapes, it may fail to accurately tag culturally specific items, resulting in unequal representation in search results.

- **Impact**: Users from underrepresented backgrounds may find that their content is less discoverable due to poor model performance on non-mainstream images.
- **Solution**: To mitigate this, the model should be trained on a more diverse dataset, and regular audits of the model's performance on varied image types should be conducted to detect any disparities.

4. Finally, there are potential **privacy risks** associated with automatically processing user-uploaded images. Users may unintentionally upload sensitive or private images, which could be analyzed without their explicit consent.

- **Impact**: This could lead to privacy violations, especially if the platform stores or shares detailed descriptions of sensitive images.
- **Solution**: Providing clear privacy policies and allowing users to **opt-out** of automatic image analysis can mitigate these risks. Additionally, ensuring compliance with data protection regulations (e.g., GDPR) is crucial for protecting user privacy.

While machine learning introduces valuable features, it also poses risks related to bias, fairness, accuracy, and privacy. These harms can be mitigated through bias detection, user validation, model retraining, and strong privacy safeguards, ensuring a more equitable and secure platform.

## Production Challenges

Deploying the **Alternative Text Generation** and **Image Search by Tags** features at scale introduces key challenges, especially in the areas of **scalability**, **cost**, and **performance**. Addressing these challenges is crucial for the system to efficiently serve millions of users.

**1. Scalability**

As the number of users grows, the volume of API requests to Azure's Computer Vision service will increase significantly. With millions of image uploads, there's a risk of API throttling or performance delays due to the high volume of requests.

- **Problem**: API rate limits could slow down the system, leading to delays in generating alt text and tags.
- **Solution**: Implementing **caching** and **queueing systems** would help manage API requests more efficiently. Caching results for similar images would reduce unnecessary calls, while queueing requests would allow the system to process uploads asynchronously, ensuring better scalability.

**2. Operating Costs**

Frequent API calls to Azure can lead to high operating costs, particularly with a large user base. Each image upload triggers multiple requests for generating descriptions and tags, which could become expensive.

- **Problem**: Uncontrolled API costs could exceed the platform's revenue or budget.
- **Solution**: Use **local models** to handle some requests instead of always relying on the Azure API, reducing costs. Additionally, setting **thresholds** for API usage, such as limiting calls based on image size or type, can further control operating expenses.

**3. Performance and Latency**

High volumes of API calls and database queries for tags and descriptions could result in latency, affecting the user experience during image uploads and searches.

- **Problem**: Increased latency can slow down uploads and search functionality, leading to user frustration.
- **Solution**: Use **asynchronous processing** to handle image uploads and API calls in the background. Optimizing the database with **indexing** for faster tag-based search queries can reduce latency and improve response times.

**4. Data Storage**

As the platform scales, storing millions of images, descriptions, and tags becomes a challenge, both in terms of performance and cost.

- **Problem**: Large amounts of stored data can lead to slower queries and higher storage costs.
- **Solution**: Use **cloud storage** for images (e.g., AWS S3) and relational databases for metadata. **Database partitioning** can also help distribute the load and improve query performance.

Key production challenges include managing scalability, controlling costs, and optimizing performance. Solutions such as caching, asynchronous processing, local models, and rate-limiting can help ensure the platform remains efficient and scalable, even with millions of users.