WEP Warwick Evans Publishing

International Journal of Computer Science and Information Technology

ISSN: 3005-9682 (Print), ISSN: 3005-7140 (Online) | Volume 7, Number 3, Year 2025
DOI: https://doi.org/10.62051/ijcsit.v7n3.05
Journal homepage: https://wepub.org/index.php/IJCSIT/index

Automated InChI Sequence Generation from Chemical Images with Configurable Encoder-Decoder Architecture

Suxuan Liu

Faculty of Science and Technology, Beijing Normal-Hong Kong Baptist University, Zhuhai, 519087, China

ABSTRACT

In the chemical and pharmaceutical domains, researchers frequently rely on structural diagrams found in patents, scientific literature, and archival documents. Converting these diagrams into standardized text representations, such as the International Chemical Identifier (InChI), is traditionally a labor-intensive manual process. To address this challenge, we introduce an optical chemical structure recognition (OCSR) system based on an encoder–decoder framework that translates molecular images into InChI strings efficiently and accurately. The workflow begins with image preprocessing, where each structural diagram is resized and transformed into normalized tensors. The encoder then extracts salient visual features through backbone networks, including ResNet, EfficientNet, and Vision Transformers, generating high-dimensional feature maps that capture both spatial and semantic information. These are passed to the decoder, implemented with GRU, LSTM, or Transformer architectures, which sequentially outputs InChI tokens. To enhance performance, the framework integrates a soft attention mechanism that directs focus toward relevant image regions during decoding, and employs a gradually reduced teacher forcing strategy to improve robustness during training. We evaluate multiple architectural pairings on a large-scale dataset, benchmarking model outputs against ground-truth InChI strings using exact match accuracy, Levenshtein distance, BLEU, and METEOR metrics. Results reveal clear performance variations across encoder–decoder configurations, with EfficientNet-B0 combined with an LSTM decoder achieving accuracy up to 84.0% on a 20K dataset, striking an effective balance between computational efficiency and predictive precision. These findings provide critical insights into designing OCSR systems tailored for varying levels of chemical complexity and resource availability, thereby advancing automated molecular translation for diverse applications in research and industry.

KEYWORDS

Optical Chemical Structure Recognition; InChI; Image Captioning; Attention Mechanism; Encoder-Decoder Ar- chitecture; Deep Learning

1. INTRODUCTION

Image captioning has seen substantial progress in recent years, with leading models achieving impressive results on benchmarks like MS-COCO and demonstrating practical ap- plications across various domains [1]. As illustrated in the example image (Figure 1), modern image captioning systems can successfully generate descriptive text for diverse scenarios, from "a dog lying on its back in the sand" to "a little boy going down a blue slide", by recognizing visual elements and translating them into coherent natural language descriptions. This success largely stems from architectures employing self- attention mechanisms that effectively bridge visual features and linguistic representations. However, while general image captioning continues to advance, specialized domains like molecular structure translation introduce fundamentally dif-ferent challenges. The BMS

Molecular Translation task, for instance, involves converting chemical structure images into International Chemical Identifier (InChI) strings. This process presents particular difficulties due to the complex relationship between visual molecular representations and their precise textual encodings [2]. Unlike the natural images shown in Figure 1, where approximate descriptions may suffice (e.g., a "brown dog" versus a "spotted dog"), molecular translation demands exact character-level accuracy to maintain chemical validity.

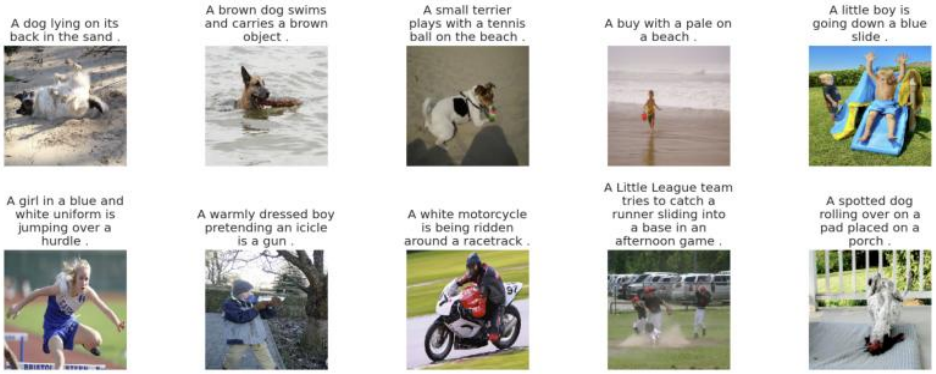

Figure 1. Image and description

Central to this translation task is the International Chemical Identifier (InChI), a standardized textual representation used to describe chemical molecules and structures. InChI is both human-readable and machine-processable, offering a concise and easily convertible digital format compared to traditional chemical structure images, thereby facilitating data storage and sharing. The InChI code itself consists of a series of layers separated by slashes (/), starting with the version number, followed by the main layer containing basic molecular information [3]. Subsequent layers provide further details, including molecular formula, atomic bonding, fixed hydrogen atoms, charge state, stereochemical information, and isotopic data. Each compound possesses a unique InChI, making it functionally similar to the IUPAC name for identifying chemical structures. Notably, unlike SMILES, InChI is non- proprietary and freely accessible, with its development sup- ported by a non-profit organization [2]. Consequently, InChI has become the widely accepted standard for representing chemical structures in digital formats.

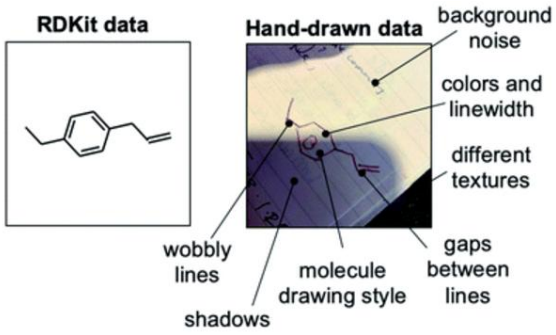

Figure 2. RDKit compare with Hand-draw

The BMS molecular translation task, however, presents distinct technical challenges that extend beyond those of con-ventional image captioning. Firstly, chemical structure images exhibit significant variations in representation style, resolution quality, and drawing conventions across different sources; studies have shown up to 42% structural variance between publishers [3]. Furthermore, handwritten chemical structures add another layer of complexity due to diverse writing formats, habits, and font styles. Figure 14 illustrates this disparity: the computer-generated standard pattern from RDKIT DATA [4] (left) is markedly clearer and cleaner than a typical handwrit- ten example (right). Secondly, molecular structures contain critical spatial information that must be precisely captured. This includes stereochemical features—often represented by subtle visual cues like dashed or wedged bonds—that deter- mine three-dimensional arrangements and biological activity. Moreover, InChI strings for drug-like compounds average 80-120 characters in length,

substantially longer and more unpredictable than typical image captions. This characteristic poses significant hurdles for sequence generation models, which are prone to error accumulation when producing such extended and precise textual outputs. Indeed, analysis of existing systems reveals error rates that increase exponentially with molecular complexity, with accuracy dropping from 94% for simple structures to below 30% for compounds with multiple stereogenic centers [3]. The translation process also necessitates chemical knowledge integration, as minor visual misinterpretations can yield chemically invalid or radically different molecular specifications.

In chemical research and industrial applications, the ac-curate representation and efficient conversion of chemical structures are crucial. Currently, however, generating InChI sequences predominantly relies on manual input or extraction from known chemical structure files (such as MOL files). This approach is not only inefficient but also susceptible to errors stemming from human oversight or incompatible data formats. Compounding this issue, a vast amount of chemical information exists in image form—including hand-drawn di-agrams, scanned literature illustrations, and images in online databases—which cannot be directly recognized and utilized by existing InChI generation tools. Therefore, developing methods to automatically, efficiently, and accurately generate InChI sequences from chemical images has become an urgent problem in chemoinformatics.

Despite significant advances in the field, from early rule-based systems to modern deep learning approaches, chemical structure recognition remains a formidable challenge. This is particularly true for complete InChI generation, where accuracy rates for complex structures often remain below 30% in practical applications [1]. This persistent gap underscores the need for more robust architectural approaches to address the unique challenges of molecular translation.

To address these challenges, this paper makes several key contributions:

1) We provide a comprehensive architectural analysis of 15 different encoder-decoder combinations for molecular structure translation, systematically evaluating five en-coder variants (ResNet18, ResNet34, EfficientNet, Swin Transformer, and ViT) paired with three decoder archi-tectures (GRU, LSTM, and Transformer).

2) We demonstrate that, for molecular image translation using smaller datasets, CNN-based encoders paired with RNN-based decoders consistently outperform pure Transformer architectures, while also maintaining faster inference times. This finding offers practical insights, particularly for applications with limited dataset sizes.

3) We enhance model performance through a graduated teacher forcing strategy and an attention mechanism that enables the model to focus on relevant molecular substructures during sequence generation.

Our experimental results provide practical architectural guidance for researchers seeking to balance accuracy, com- putational efficiency, and chemical validity in the automated generation of InChI strings from structural images.

2. LITERATURE REVIEW

The core of the chemical molecular translation, from chem- ical structure diagrams to SMILES or InChI [3] formats, is a key task in chemistry and pharmaceuticals. With over 108 million chemical compounds registered in CAS Registry as of 2023, efficient molecular translation systems are essential for drug discovery and chemical database curation. The evolution of this field has progressed from rule-based approaches to deep learning technologies, with models including OCSR, Trans-formers [5], and optimization index, progressively improving translation accuracy from below 60% to over 90% on standard benchmarks.

2.1. Optical Chemical Structure Recognition

Optical Chemical Structure Recognition(OCSR) is a tech- nology that automatically extracts chemical structure informa- tion from images of chemical structures and converts it into computer-readable forms such as InChI or SMILES. OCSR mainly consists of two approaches: rule-based and traditional image processing methods, and the method based on a data- driven deep neural network. Most OCSR methods follow a rule-based approach. After the feature vectors are extracted, the vectors and nodes are interpreted as bonds and atoms. In the early days, software such as Optical Structure Recogni- tion Software to Recover Chemical Information (OSRA) [6], Kekule [7], and CliDE [8] were developed, which can convert chemical molecule images into computer-readable sequence forms based on rules.

Kekule [7] is the first fully developed OCSR tool, used for processing such as scanning, optical character recognition, and graphic editing. Furthermore, OSRA [6] is an open-source tool developed by NCI that can directly convert extracted chemical structures into SMILES or SDF formats. However, it cannot identify charges or isotopes. CliDE [8] has further improved the previous method, supporting the extraction of chemical structures from images and the automatic detection of incorrect expressions, which greatly enhances the accuracy and flexi- bility of chemical structure recognition. However, these rule-based systems have certain drawbacks. When the molecular diagram is blurred or has an unclear representation, it is difficult for the system to give reasonable prediction results. This is mainly because the various recognition components of rule-based systems are interdependent, which makes it difficult to enhance their performance further.

OCSR plays a crucial role in many branches of chemistry, such as synthetic science, natural product research, and drug discovery. However, traditional rule-based methods are ineffi-cient and have a high error rate [6]. In recent years, the OCSR method based on deep learning has been proposed. Unlike rule-based methods, deep learning-based methods can identify chemical structures without hard-coded rules. For instance, in 2019, Staker et al. proposed the first OCSR method based on deep learning, which achieved an accuracy rate of up to 83% on the validation set [9]. On this basis, subsequent deep learning models were continuously optimized. Among them, DECIMER 1.0 [10], with its advanced architecture de-sign, successfully achieved a Tanimoto similarity level of approximately 96% in a large dataset containing 30-35 million molecules, significantly improving the accuracy and efficiency of chemical structure recognition.

2.2. CNN and CNN-LSTM Models for OCSR

In the field of image recognition, convolutional neural networks (CNNs) are a fundamental and powerful technology with remarkable achievements in multiple fields. CNN [11, 12] is capable of capturing the spatial hierarchy and patterns in images and is suitable for handling tasks with structured visual data, such as the analysis of complex chemical structures.

1) ChemGrapher: ChemGrapher is a modular CNN model specifically designed for OCSR [13]. It contains two main modules: the segmentation module and the position recog-nition module. The segmentation module adopts the dilated convolution technology, enabling the model to expand the size of the receptive field without increasing the number of parameters, thereby being able to capture the global context information more efficiently. The output of the segmentation module is then sent to the classification module. First, lo-cate and identify the atoms in the picture. Subsequently, the connection relationship between atoms is further analyzed to identify the existence and type of chemical bonds. This phased processing approach enables ChemGrapher to analyze com-plex chemical structures with more accuracy. Compared with the traditional Non-deep Learning Neural Network (OSRA), the Error Percentage has decreased from 31.3% to 24.6% [13].

2) CNN-LSTM: Immediately after the emergence of Chem-Grapher, ChemPix[14] came into being. It is a model that uses deep learning to identify chemical structures. The NN architecture it adopts

consists of a convolutional neural net- work (CNN) encoder and a long Short-Term memory (LSTM) decoder with cluster search and attention mechanisms. In the encoder, convolutional filters are applied to the image, and the results are passed to the next hidden layer. Meanwhile, it can also retain the spatial positional relationship between pixels. In the decoder, a recurrent neural network is applied to learn the relationship between the beginning and the end of a string.

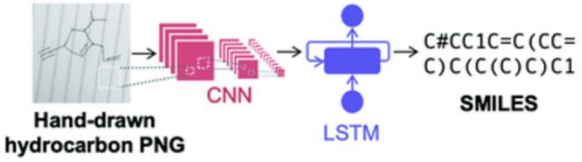

Figure 3. CNN-LSTM framework

2.3. Hybrid CNN-Transformer Models for OCSR

The success of the Transformer architecture in the NLP field has driven its application in molecular translation, as it can better capture long-range dependencies in complex molecular structures [15-18]. The core of Transformer's effectiveness lies in its self-attention mechanism, which com-putes attention scores between elements using Query (Q), Key (K), and Value (V) matrices according to:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{1}$$

Where d_k represents the dimension of the key vectors. This formulation allows the model to dynamically focus on relevant molecular substructures during translation, crucial for accurate InChI generation.

1) DECIMER1.0: The DECIMER [19] platform adopts Inception v3/efficientnet-B3 as the encoder and transformer as the decoder to convert the chemical structure images into SMILES. Compared with the RNN-based method, an accuracy rate of up to 90.3% was achieved on the PubChem dataset, and the inference time was shortened by 42%.

2) MolNexTR: In addition, the MolNexTR model [20], by combining ConvNeXt CNN and Vision Transformer, further improves its robustness to diverse drawing styles, reaching 81-97% accuracy across multiple test sets. While Transform-ers excel at capturing global molecular features, their node-agnostic processing can still miss certain structural nuances, leading researchers to explore graph-based approaches that explicitly model molecular topology.

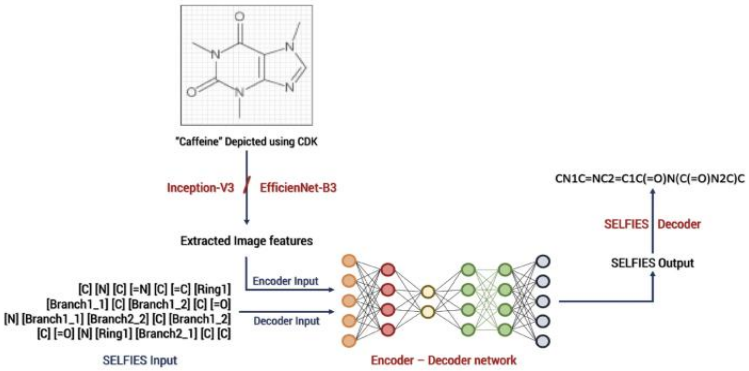

Figure 4. DECIMER framework

The above are some models that combine the basic trans- former. In recent years, with the increasing demand for visual tasks, an improved version of the transformer structure has emerged. The swin-transformer [21] is currently a relatively advanced architecture. The Swin Transformer introduces the

"inductive bias" of CNN by introducing methods such as im-age layering, local window attention, and translation Windows, significantly improving the performance of the Transformer in visual tasks and simultaneously reducing the computational cost. Therefore, we choose Swin Transformer as the important encoder-decoder of our OCSR.

2.4. Data Enhancement and Pre-Training Strategies

Most public data sets are too small to support modern deep learning models, with typical molecular image datasets con-taining only 10,000-50,000 samples compared to the millions of examples in standard computer vision benchmarks. Histori-cal data sources typically have some level of image corruption. In order to solve this problem, researchers have proposed a va-riety of data enhancement techniques. The RanDepict tool [22] improves the model's ability to generalize over the test set by randomizing the drawing parameters of the chemical structure (e.g., font, color, etc.) to produce a diverse composite image, generating over 1 million synthetic variations from just 50,000 base molecules. This approach reduced overfitting by 23% in subsequent models. In addition, MolParser [23] also improves the translation ability of complex chemical structures, such as Markush structures by pre-training on large-scale synthesized data sets (2.4M images) and then fine-tuning on real set data, achieving state-of-the-art results with 95.7% accuracy on the Pistachio dataset. These data-centric approaches complement architectural innovations, highlighting the dual importance of model design and training methodology.

Despite recent progress, molecular translation still faces several challenges. Future research should focus on two promising directions: developing more efficient models, such as MolScribe [24], which optimizes computational resources while maintaining high accuracy; and incorporating domain knowledge, as demonstrated by OCMR [25], which enhances chemical plausibility through symbolic reasoning. These ap-proaches will advance molecular translation toward the ulti-mate goal of seamless conversion between visual and textual chemical representations.

3. METHODOLOGY

3.1. Workflow Description

Figure 5 illustrates the overall workflow of our system. The workflow is designed as an end-to-end encoder-decoder architecture in which molecular images undergo several mod-ularization stages to generate structured textual representations of their chemical formulas.

The process begins with Raw Data Loading, where meta-data files containing molecular identifiers and their corre-sponding InChI strings are loaded and combined with paths to associated molecule images. This forms the base dataset used for training and validation.

Next, during the Image Reading and Preprocessing stage, raw molecular images undergo several transformations. They are first resized to a fixed resolution of 256×256 pixels, then normalized to standardize the pixel value distribution. Finally, the images are converted into tensor format using the PyTorch ToTensor() transformation for compatibility with the neural network input layer.

In parallel, target InChI strings are processed through InChI Tokenization. Rather than using complete InChI descrip- tors, we extract only the molecular formula component (e.g., InChI=1S/CH4/h1H4 → CH4) to simplify the sequence learning task. These simplified InChI strings are then tokenized at the character level, with each character mapped to a unique integer ID, including special tokens <sos>, <eos>, and <pad> for sequence control.

Once the preprocessing is complete, the image tensor is passed into the Encoder Module, where visual features are extracted using backbone architectures such as ResNet18, ResNet34, EfficientNet, or

Vision Transformers. The encoder transforms the input image into a high-dimensional feature tensor, typically of shape [batch_size, 64, 512].

Subsequently, these extracted features are consumed by the Decoder Module, which is responsible for generating the target sequence in an autoregressive manner. At each decoding time step, the decoder receives both the encoded image feature along with the previous character (ground truth during training via teacher forcing, and predicted token during inference). We experiment with different decoding modules, including GRU, LSTM, and Transformer-based architectures.

Finally, in the Evaluation stage, the predicted token se-quences are converted back to InChI strings and compared to the ground truth using accuracy and character-level metrics such as Levenshtein distance. The model is trained using cross-entropy loss, with padding tokens ignored to avoid penalizing unequal-length sequences.

This modular and flexible pipeline allows for experimenta-tion with different combinations of encoders and decoders, making it suitable for evaluating the impact of vision and sequence modeling components on molecular text generation performance.

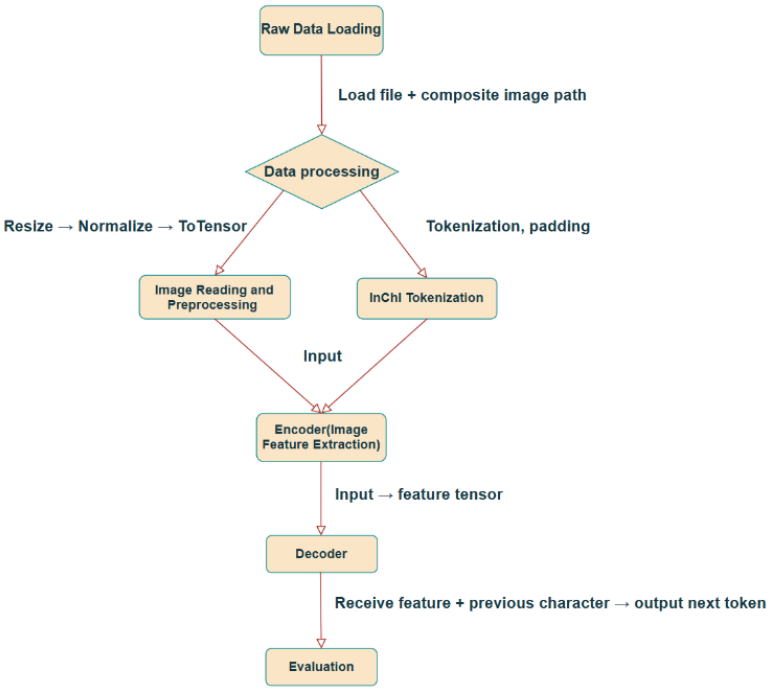

Figure 5. BMS Molecular Translation Flow Chart

3.2. Dataset Description

To train and evaluate our encoder-decoder architecture, we require high-quality molecular structure images paired and their corresponding InChI representations. The dataset used in this study is sourced from the Kaggle BMS Molecular Translation competition [26]. It comprises grayscale or RGB 2D chemical structure images along with IUPAC-standardized InChI strings as chemical identifiers.

Each data sample contains two primary fields:

• Image Identifier (image id): A unique alphanumeric string that corresponds to the filename of a molecule image.

• Chemical Identifier (InChI): A textual delineation of a molecule's structure, encompassing elements such as version designation, molecular formula, and structural particulars.

The InChI string customarily incorporates three segments:

• Version: Indicates the InChI standard version, e.g., InChI=1S.

• Formula: Represents the molecular formula, e.g., C2H6O.

• Structure: Encodes detailed structural information.

To visualize and examine the data, an example of a molec-ular image is shown in the following image.

This is the corresponding InChI formula:

InChI=1S/C13H20OS/c1-9(2)8-15-13-6-5-10(3)7-12 (13)11(4)14/h5-7, 9, 11, 14H, 8H2, 1-4H3

3.3. Data Preprocessing

Following the dataset description, we now detail the prepro- cessing steps required to prepare the data for our deep learning model. To ensure compatibility with the neural network archi- tecture, we performed several operations on the raw dataset, including image conversion, disambiguation of InChI strings, and batch construction of image paths.

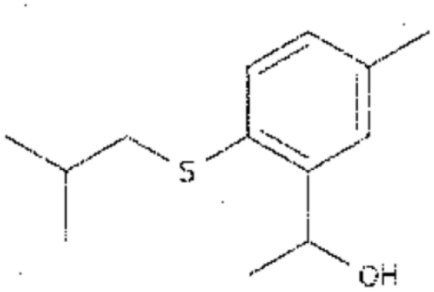

Figure 6. Chemical structure image

1) Image Path Generation: Each molecular image contains a unique image ID. To facilitate file loading, we generate a file path to the image based on the first three characters of the image ID. In the preprocessing stage, the image paths are dynamically reconstructed programmatically to efficiently locate and load the corresponding image files, thus facilitating subsequent data loading and processing.

2) InChI Segmentation: Once the image paths are es-tablished, we process the associated InChI strings. For each sample, the full InChI string is parsed into three parts:

• Version prefix: Indicates the version information of the chemical identifier.

• Molecular formula: Indicates the molecular formula of the chemical.

• Structural data: Indicates the detailed structural infor-mation of the chemical substance.

The length of each InChI formula is also computed for later analysis of data consistency and sequence modeling.

3) Image Processing: After organizing the dataset struc-ture, we focus on preparing the image content. Each image is preprocessed using OpenCV and the Albumentations library:

• Resize: All images are resized to 256 × 256 pixels.

• Normalization: Pixel values are scaled from [0, 255] to [0, 1].

• ToTensor: The images are converted to PyTorch compat-ible tensors.

This ensures consistent shape, format, and intensity distribu-tion across all samples.

4) Formula Character Encoding: In parallel with image processing, we prepare the target sequences. The molecular formulas are tokenized at the character level. A vocabulary is built from all unique characters in the dataset, with each character assigned a unique index. Special tokens such as <sos>, <eos>, and <pad> are added for sequence control. The formulas are then converted into numerical sequences to be used as decoder input/target.

5) Dataset Splitting and Batching: With both image and formula data properly processed, we prepare the final dataset structure for training. To enhance the generalization ability of the model, we randomly split the dataset into a training set and a validation set with a ratio of 90% and 10%. To improve training efficiency, we use PyTorch's DataLoader to batch load the dataset. In each batch, padding operations are automatically performed to align target sequences of different lengths to ensure the consistent shape of the labeled data within the batch, and a function is used to handle variable- length sequences to ensure the stability of the data fed into the model.

3.4. Encoder-Decoder Architecture

Our molecular captioning system employs a flexible encoder-decoder framework, taking a 2D molecular image as input and generating an InChI sequence as output. The encoder extracts high-level visual features from the image via a configurable backbone, while the decoder produces the sequence in an autoregressive fashion.

As illustrated in Figure 7, the image is first passed into the encoder module, which can be instantiated using various ar-chitectures, including ResNet18, ResNet34, EfficientNet, Swin Transformer, or ViT. The encoder generates a feature tensor of shape [batch, 64, 512], capturing both spatial and semantic features.

This feature is subsequently projected through a linear layer to align with the decoder's expected input dimensions. The decoder, which can be a GRU, LSTM, or Transformer, produces the InChI token sequence step by step, conditioned on both the encoder features and the previously generated tokens.

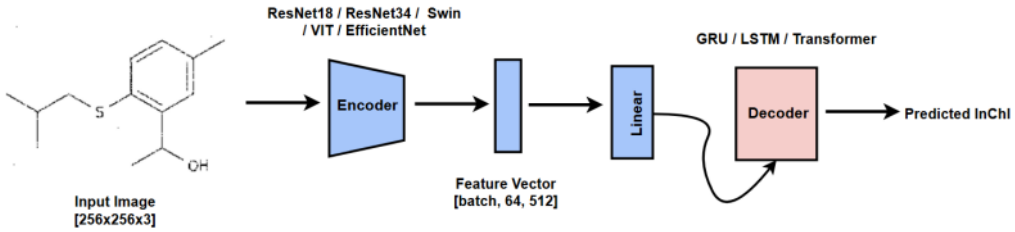

Figure 7. Encoder-Decoder Architecture

The Attention mechanism has been extensively utilized in neural networks, particularly in sequence generation tasks. A key reason lies in its ability to enable the model to focus on specific parts of the input sequence when generating each output token, which is vital for capturing dependencies between distant elements within the sequence, a challenge for traditional models such as RNNs.

In this project, the Attention mechanism is integrated into both the encoder and the decoder. This integration aims to enhance the modeling of connections between image features and sequence generation.

• Input Specification: The Attention mechanism receives two inputs: the image feature representation from the encoder (e.g., [batch_size, 64, 512]) and the target sequence (e.g., [batch_size, seq_len]). These inputs are processed to learn the relationships between different parts of the sequence.

• Feature Extraction: The Attention mechanism computes a weighted sum of input features based on their relevance to each position in the sequence. This is done by calcu-lating attention scores through a similarity function (e.g., dot-product or scaled dot-product) between the query (target sequence) and key (image feature). These scores determine how much focus should be placed on each part of the input sequence for generating the next token in the output. The output dimension after applying Attention is [batch_size, seq_len, vocab_size].

3.5. Encoder Module

Building upon the general architecture described above, we now detail the specific encoder implementations. To identify the optimal model architecture, this study employs multiple Encoder and Decoder modules for feature extraction and sequence modeling, respectively. The Encoder part includes multiple visual Transformer architectures and Convolutional Neural Networks (CNNs). Each Encoder receives input as a molecular structure map of size [3, 256, 256] and outputs high-dimensional image features for decoder processing. The Decoder modules, on the other hand, includes GRU, LSTM, and Transformer models based on the self-attention mecha- nism. The specific implementation of each encoder variant is described below.

1) ResNet: ResNet (Residual Networks) is an architecture based on deep convolutional neural networks, which uses residual connections to effectively solve the problem of gradi-ent disappearance in deep networks. For this project, ResNet-34 is selected as the base Encoder. By removing the last two layers of ResNet, which are the average pooling layer and the fully connected layer, it is used as a feature extractor.

• Input specification: The input image size is [batch_size, 3, 256, 256].

• Feature extraction: ResNet extracts spatial features of images through convolutional layers, and the out-put feature dimension is [batch_size, 512, 8, 8]. The features are then rearranged and flattened to [batch_size, 64, 512], suitable for connecting with the Decoder.

2) Efficient-Net: Efficient-Net adjusts the depth, width, and resolution of the network through Compound Scaling to optimize performance and computational efficiency in multi-ple dimensions. In this project, EfficientNet-B0 removes its classification layer to facilitate image feature extraction.

• Input specification: Input image size is [batch_size, 3, 256, 256].

• Feature extraction: After the feature extraction part of Efficient-Net, the output dimension is [batch_size, 512, 8, 8] and adjusted to [batch_size, 64, 512] by linear projection.

3) Swin-Transformer: Swin Transformer is a window-based self-attention mechanism designed for efficient process- ing of large-scale image data. In this project, SWIN-T (Swin Transformer Small version) is used as an Encoder to directly obtain image features by removing classification headers.

• Input specification: The input image size is [batch_size, 3, 256, 256].

• Feature extraction: Through the feature extraction mod- ule of Swin Transformer, the output feature dimen- sion is [batch_size, 768], and then adjusted to [batch_size, 64, 512] by linear projection.

4) Vision Transformer (ViT): The Vision Transformer (ViT) directly divides the image into small patches for model- ing using the standard Transformer self-attention mechanism. The ViT-B/16 is selected as the Encoder in order to remove the classification header and output the global features of the image directly.

• Input specification: The input image size is [batch_size, 3, 256, 256].

• Feature extraction: After the processing from the ViT model, the output feature dimension will be [batch_size, 768], and then adjusted to [batch_size, 64, 512] through the projection layer.

3.6. Decoder Module

With the molecular structure features extracted by the encoder, the decoder is responsible for generating the InChI sequence autoregressively. It receives the image features from the encoder and previously generated tokens to predict the next token in the sequence. We implemented three decoder variants to compare performance across different sequence modeling approaches.

1) Gated Recurrent Unit (GRU): The GRU is an improved RNN architecture that can control the flow of information through reset and update gates to capture long-term dependen- cies in sequence modeling. In this project, the GRU Decoder will be used to generate sequences based on image features.

• Input specification: The input is the feature from the Encoder [batch_size, 64, 512], and the target sequence [batch_size, seq_len].

• Output: The decoder outputs the prediction results of [batch_size, seq_len, vocab_size].

2) Long Short-Term Memory (LSTM): LSTM captures long-term dependencies via input gate, forget gate, and output gate mechanisms. We use LSTM as a Decoder to generate sequence output with long-term dependent information by combining it with the self-attention mechanism.

• Input specification: Similar to the GRU Decoder, the input includes image features of [batch_size, 64, 512] and target sequence [batch_size, seq_len].

• Output: The Decoder generates prediction results with a dimension of [batch_size, seq_len, vocab_size].

3) Transformer Decoder: Transformer Decoder is based on a self-attention mechanism and is widely used for sequence generation tasks. We combine the Transformer decoder with the encoder output for autoregressive generation.

• Input specification: The input is the image feature [batch_size, 64, 512] from the Encoder, and the target sequence [batch_size, seq_len].

• Output: Decoder outputs the prediction results of [batch_size, seq_len, vocab_size].

3.7. Loss Function

To optimize the model during training, we adopt the cross-entropy loss function, which is widely used in sequence prediction tasks. At each decoding step, the model predicts the probability distribution over the vocabulary for the next character. The loss function compares the predicted distribu-tion against the ground truth token and penalizes incorrect predictions.

Let y_t denote the ground truth token at time step t, and let $\hat{y}_t$ be the predicted probability distribution. The cross-entropy loss at each step is defined as:

$$\mathcal{L}_{\text{CE}}(y_t, \hat{y}_t) = -\sum_{i=1}^{V} \mathbf{1}_{\{i=y_t\}} \log(\hat{y}_t^{(i)}) \tag{2}$$

Probability for token i at step t.

During training, we compute the average loss across all time steps and all samples in the batch. Additionally, we apply a padding mask to ignore the '<pad>' tokens in the target sequence so that they do not contribute to the final loss value. This ensures that only meaningful tokens affect the model optimization process.

The total loss over a sequence of length T is:

$$\mathcal{L} = \frac{1}{T} \sum_{t=1}^{T} \mathbf{1}_{\{y_t \neq \langle \text{pad} \rangle\}} \cdot \mathcal{L}_{\text{CE}}(y_t, \hat{y}_t) \tag{3}$$

3.8. Optimization

The key to effective model convergence lies in the op-timization of neural network training. This section details the parameter settings used in training, including parameter tuning, teaching force, and other optimization strategies.

1) Learning Rate and Scheduler: We use the Adam optimizer for deep learning model training, which is well-suited to handle sparse gradients and large parameter spaces. The hyperparameters used for fine-tuning are as follows:

• Learning rate (lr): The initial learning rate is set to 1×10^{-4}, a common choice for fine-tuning deep models. This relatively high learning rate facilitates faster initial convergence.

• Weight decay: To prevent overfitting and encourage generalization, weight decay (L2 regularization) with a value of 1×10^{-6} is applied to both the encoder and decoder.

• Cosine annealing scheduler: To balance convergence speed and training stability, we use a cosine annealing scheduler. This scheduler gradually reduces the learning rate after each cycle, enabling finer-grained updates as the model approaches convergence.

These choices enable a balance between convergence speed and training stability. The encoder and decoder are trained using separate optimizers. The encoder is initialized by pre-trained weights and fine-tuned at a relatively low learning rate to preserve useful visual representations. In contrast, the decoder is trained with a slightly higher learning rate in order to facilitate learning how to map feature vectors to labeled sequences from scratch.

2) Teacher Forcing with Gradual Reduction: Teacher forc-ing is a training technique where, during sequence generation, the model receives ground truth tokens as input rather than its own predictions. While this approach accelerates training convergence, it creates a discrepancy between training and in-ference conditions known as exposure bias. During inference, the model must rely on its own predictions, which can lead to error accumulation when predictions deviate from the expected distribution.

To address this challenge, we implement a teacher forcing strategy with gradual decay. Rather than using a fixed teacher forcing ratio throughout training, we gradually reduce the dependency on ground truth inputs:

• Early stages (high teacher forcing): In the early training phase, a high teacher forcing ratio is applied so that the model learns the correct sequence structure from the ground-truth outputs.

• Later stages (low teacher forcing): As training pro-gresses, the ratio is gradually reduced, encouraging the model to rely more on its own predictions. This helps the model adapt to inference-time conditions where ground-truth tokens are not available.

This gradual transition enhances the robustness of the model and reduces the discrepancy between training and inference behavior, mitigating error propagation during sequence gener-ation.

Algorithm 1: Decoder Training with Teacher Forcing

Input: Target sequence $Y = \{y_1, y_2, ..., y_T\}$, encoded features F, teacher forcing ratio r
Output: Predicted token sequence $\hat{Y}$

Initialize $input \leftarrow \langle sos \rangle$;
for $t \leftarrow 1$ **to** T **do**
  $output \leftarrow \text{Decoder}(input, F)$;
  $\hat{y}_t \leftarrow \arg\max(output)$;
  Sample $p \sim \mathcal{U}(0, 1)$;
  if $p < r$ **then**
    $input \leftarrow y_t$; // Use ground truth (teacher forcing)
  else
    $input \leftarrow \hat{y}_t$; // Use model prediction
return $\hat{Y}$

The code snippet demonstrates how the model's robustness can be improved by a pedagogical forcing mechanism that guides the model from learning with real data to gradually transitioning to learning with self-generated predictions.

3) Optimizers and Fine-tuning: The Adam optimizer is employed for both encoder and decoder due to its effective-ness with sparse gradients and large parameter spaces. The following adjustments are implemented:

• Encoder: The encoder undergoes fine-tuned with a rel-atively low learning rate to enable effective learning of feature representations from the pre-trained weights.

• Decoder: The decoder is trained with a slightly higher learning rate, which allows it to quickly learn the mapping from encoder features to the target sequence.

With the training methodology established, we now require robust quantitative measures to objectively assess how effec-tively each model configuration translates molecular images to InChI strings, guiding our architectural decisions based on empirical evidence rather than theoretical assumptions.

3.9. Evaluation Metrics

Evaluating molecular image-to-InChI translation presents unique challenges distinct from general image captioning tasks. Chemical formulas demand character-level precision, as even minor errors can result in completely different com-pounds with vastly different properties. We selected a com-plementary suite of metrics that specifically address these challenges and provide a comprehensive assessment of our encoder-decoder architectures.

1) Exact Match Accuracy: Given the critical importance of chemical accuracy in molecular applications, we prioritize exact match accuracy as our primary metric. Unlike general image captioning where approximate descriptions may suffice, InChI translation requires perfect character-level precision to maintain chemical validity.

$$\text{Accuracy} = \frac{\text{Number of exact matches}}{\text{Total predictions}} \tag{4}$$

This strict binary metric guided our architecture selection process, revealing that the ResNet-34 and EfficientNet-B0 encoders consistently produced the most chemically accurate predictions across our test compounds.

2) Average Edit Distance: Since molecular formulas can share many characteristics even when not perfectly matching, we employ edit distance to capture partial correctness. This metric is particularly valuable for molecular translation as it quantifies how many character-level operations would be needed to correct imperfect predictions.

$$\text{Edit Distance (A, B)} = \text{min edits to convert A} \rightarrow \text{B} \tag{5}$$

For example, if our model predicts "C6H5Cl" when the target is "C6H4Cl2", an edit distance of 1 indicates a near-miss that might still be useful in chemical search applications. This metric revealed that our EfficientNet-B0 with LSTM config-uration achieved the lowest average edit distance (0.3824) on the 20K dataset, indicating superior character-level prediction even when exact matches weren't achieved.

3) BLEU Score (BLEU-1 to BLEU-4): To evaluate n-gram overlap at different granularities, we adapted BLEU scores for character-level assessment of our molecular predictions. While traditionally used in machine translation, BLEU proves valuable for molecular sequences by capturing local character pattern accuracy.

$$\text{BLEU} = \text{BP} \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n\right) \tag{6}$$

The brevity penalty BP is defined as:

$$\text{BP} = \begin{cases} 1 & \text{if } c > r \\ \exp\left(1 - \frac{r}{c}\right) & \text{if } c \leq r \end{cases} \tag{7}$$

By examining BLEU-1 through BLEU-4 scores, we gain insights into how well our models preserve important molecu-lar substructures. For instance, higher BLEU-3 scores indicate better preservation of atomic groupings like hydroxyl (-OH) or carboxyl (-COOH) functional groups. Our best architectures achieved BLEU-4 scores exceeding 0.92, confirming their ability to maintain chemical substructure integrity.

4) METEOR Score: For molecular formulas where ele-ment order can vary while maintaining chemical equivalence, METEOR offers a more flexible evaluation by balancing pre-cision with recall and accounting for potential rearrangements.

$$\text{METEOR} = \frac{(R + 9P)}{10PR} \cdot (1 - P_m) \tag{8}$$

Where P represents precision, R represents recall, and P_m is the fragmentation penalty:

$$P_m = 0.5 \cdot \left(\frac{M}{C}\right)^3 \tag{9}$$

METEOR proves particularly valuable for evaluating our molecular translation models as it can recognize chemically equivalent representations that might appear as errors un-der stricter metrics. For example, when our model predicts "CH3OH" instead of "CH4O" (both representing methanol), METEOR assigns a higher similarity score than BLEU, ac-knowledging the chemical equivalence despite the different character arrangement.

5) Implementation and Architecture Selection: We im-plemented these metrics using the NLTK library with cus-tom adaptations for molecular evaluation. Before assessment, special tokens were removed, and results were normalized to percentage scores where appropriate. This comprehensive eval-uation framework directly informed our architecture selection process, revealing that while

Transformer-based models theo-retically offered advantages for sequence modeling, the CNN-based encoders paired with LSTM/GRU decoders consistently achieved superior performance across all metrics for molec-ular translation. This finding challenges conventional wisdom about Transformer superiority and highlights the importance of domain-specific evaluation for specialized visual-linguistic tasks like chemical structure recognition.

4. EXPERIMENTS

To systematically evaluate our architectural choices, we con-ducted comprehensive experiments across multiple encoder-decoder combinations, assessing both prediction quality and computational efficiency. This evaluation framework allows us to identify optimal architectures for different deployment scenarios and computational constraints.

4.1. Experimental Setup

1) Dataset:

• Dataset A (Small-scale):

Training set: 90,000 image–InChI pairs

Test set: 10,000 unlabeled molecular images

• Dataset B (Large-scale):

Training set: 180,000 image–InChI pairs

Test set: 20,000 unlabeled molecular images

2) Training Configuration: Due to the computational de-mands of training multiple encoder-decoder architectures, our experiments were conducted using a hybrid computing setup. Primary experiments were performed on a worksta-tion equipped with an NVIDIA GeForce RTX 4080 GPU (16GB VRAM), Intel Core i9 processor, and 64GB RAM. For more memory-intensive Transformer-based models, we sup-plemented our computing resources with cloud-based NVIDIA A100 (40GB) instances, utilizing approximately 40 hours of additional computation time. This distributed approach allowed us to efficiently evaluate all architectural combinations within our research timeline while managing computational constraints.

Our experimental strategy systematically evaluated 15 dis-tinct encoder-decoder combinations, comprising five encoder variants(ResNet18, ResNet34, EfficientNet-B0, Swin-T, and ViT-B) paired with three decoder types (GRU, LSTM, and Transformer). Each model configuration was trained for 15 epochs using the Adam optimizer with an initial learning rate of 1×10^{-4} and a cosine annealing scheduler. The training process for CNN-based encoders paired with GRU/LSTM decoders typically required 8-12 hours per combination on the RTX 4080. In contrast, Transformer-based configurations de-manded significantly more resources, requiring approximately 20-24 hours per combination on the A100 GPU. All models were implemented using PyTorch 1.10 and trained with a batch size of 32, which was adjusted to 16 for Transformer configurations to accommodate memory constraints.

Before analyzing final performance metrics, we first ex-amine the training dynamics of our models to ensure proper convergence and validate our optimization approach.

4.2. Convergence Analysis

Examining how loss values evolve during training helps us determine whether our chosen optimization approach ef-fectively suits the molecular translation task. Through the loss curves shown in Figure 8 and Figure 9, we can assess if the Adam optimizer, learning rate settings, and 15-epoch

training duration adequately serve the learning process. These trends also highlight which architectural combinations most efficiently learn to translate molecular structures into InChI strings.

The high-performing models in Figure 8 show encourag-ing convergence patterns. The ResNet34-LSTM demonstrates remarkable stability, with its loss steadily decreasing from 0.79 to 0.076 throughout the training process. Just a single validation spike appears around epoch 7, after which the model maintains consistent performance. The EfficientNet-LSTM follows a comparable smooth trajectory, although it stabilizes at a slightly higher final loss of 0.108. While ResNet34-GRU ultimately achieves comparable results, its path proves more tumultuous—validation loss occasionally jumps dramatically, particularly at epoch 6. Despite these temporary fluctuations, all these top-performing architectures successfully drive their loss values below 0.1. This confirms the effectiveness of our optimization strategy for these configurations.

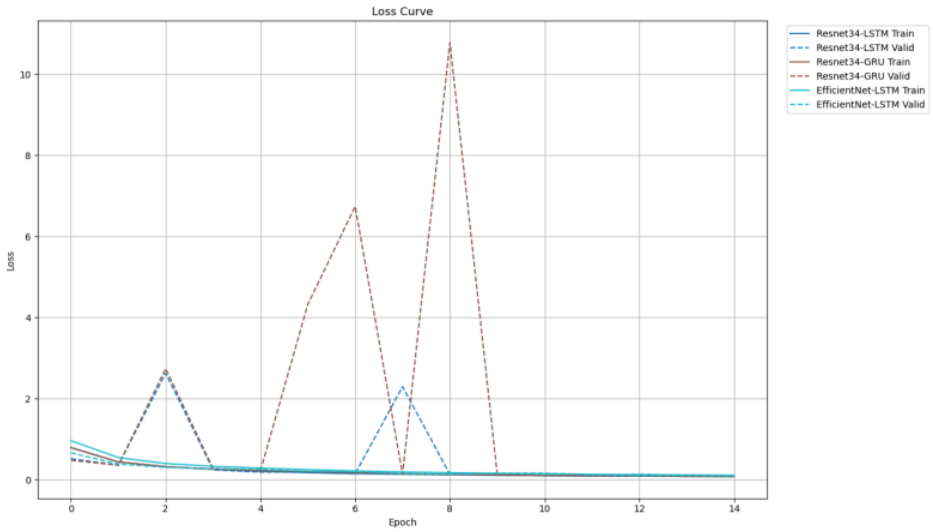

Figure 8. Train-Loss Diagram for models with highest BLEU-4 (10K Dataset)

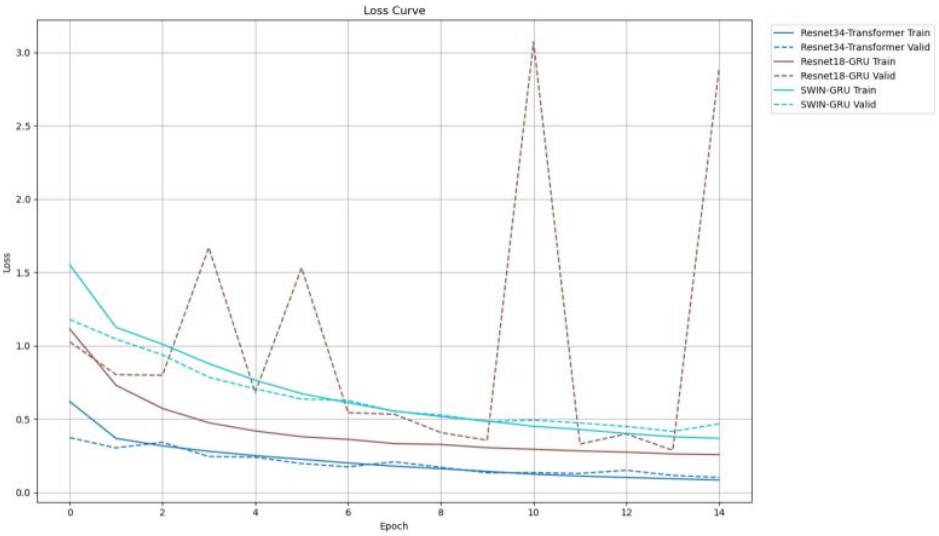

Figure 9. Train-Loss Diagram for models with Lowest BLEU-4 (10K Dataset)

In stark contrast, the lower-performing models in Figure 9 reveal significant convergence challenges. ResNet18-GRU shows significant instability throughout training, with validation loss repeatedly spiking above 2.5 in later epochs. This suggests ResNet18's feature representations may lack the depth needed for GRU to generate consistent sequence patterns. Meanwhile, ResNet34-Transformer, though more stable, plateaus at around 0.08 training loss and 0.1 validation, indicating potential

benefits from extended training. SWIN-GRU exhibits the slowest progress, only reaching a loss of 0.37 after 15 epochs—substantially higher than other models. Limited computational resources, particularly for resource-intensive Transformer models requiring nearly a day per training cycle, prevented further training extension.

It's worth noting that these convergence analyses were conducted on our 10K dataset sample as illustrated in Figures 8 and 9. When scaling to the larger 20K dataset, we observed modest performance improvements across all model configurations, with loss values typically decreasing by 0.1-0.2 times compared to the 10K training results. This improvement suggests that additional training data helps models generalize better, though the relative performance rankings between different architectures remained largely consistent across dataset sizes.

4.3. Performance Analysis

To provide a comprehensive evaluation of our architectures, we conducted extensive benchmarks across multiple metrics, with complete results available in the appendix. For detailed performance comparisons, we refer readers to Table I, Table II, and Figure 14 in the appendix, which present exhaustive metrics for all encoder-decoder combinations across both dataset sizes. These detailed analyses guided our subsequent investigation into the critical trade-off between prediction accuracy and computational efficiency in the following discussion.

When deploying machine learning models in real-world chemical informatics applications, balancing prediction quality with computational efficiency becomes critical. To identify architectures offering optimal trade-offs between accuracy and speed, we assessed each encoder-decoder combination using both prediction quality metrics and inference time. These considerations are especially important in scenarios where molecular translation systems must process large document repositories or integrate with interactive chemical drawing tools, where responsiveness is key. Figure 10 and Figure 11 visualize these performance-speed relationships, aiding in ar-chitecture selection based on specific deployment constraints.

Figure 10. Exact Match Accuracy (Bar) & Inference Time (Line) (10K Dataset)

Figure 10 shows significant variation in exact match ac-curacy across our tested architectures. ResNet-34 paired with GRU achieves the highest accuracy (76.8%) while maintaining reasonable inference speed. EfficientNet-B0 models demon-strate comparable accuracy (76.1% with LSTM, 74.2% with GRU) but with faster inference times, making them suitable for resource-constrained environments. In contrast, Transformer- based decoders paired with either ResNet variant performed poorly in exact matches despite their theoretical advantages for sequence modeling. This surprising result indicates that for molecular formula translation, the increased complexity of Transformer

architectures may actually hinder effective learning when training data is limited, as the model struggles to converge within our computational constraints.

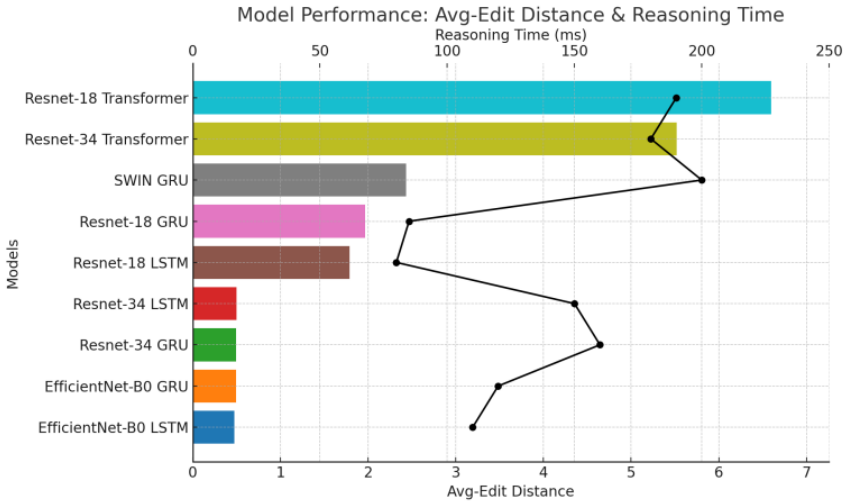

Figure 11. Average Edit Distance (Bar) & Inference Time (Line) (10K Dataset)

Figure 11 presents the average edit distance results, offer-ing a more detailed performance view by capturing partial correctness where exact matches fail. EfficientNet-B0 with LSTM achieves the lowest edit distance (0.4745), indicating it produces the most similar sequences to the ground truth when perfect accuracy isn't attained. The ResNet-34 variants follow closely with slightly higher edit distances (around 0.49). Interestingly, when considering edit distance alongside inference time, EfficientNet-based models maintain their ef- ficiency advantage without sacrificing quality. Swin-T with GRU and ResNet-18 configurations show substantially higher edit distances (1.7-2.4), suggesting these lighter architectures struggle to capture the detailed spatial features necessary for accurate molecular translation. These findings indicate that for practical deployments, EfficientNet-B0 with LSTM offers the most balanced performance profile, delivering near-optimal accuracy with the fastest inference speeds among high- performing models.

5. MODEL DEPLOYMENT

Based on our experimental results, which demonstrated promising accuracy with our best architectures achieving approximately 82% exact match accuracy on the 20K test set, we developed a lightweight application for practical molecular structure recognition. This web-based tool leverages our best-performing encoder-decoder model (EfficientNet-B0 with LSTM) to provide efficient and accurate translation of chemical structure images to InChI formulas. To enhance usability for non-expert users, we augmented the system with a large language model component that provides contextual explanations of the predicted molecules, including structural features, potential applications, and validation of the predicted formula against the visual representation.

In the initial user interface (Figure 12), researchers can upload molecular structure images through a simple drag-and-drop area or file selection dialog. The clean and intuitive in-terface emphasizes ease of use while maintaining professional aesthetics suitable for laboratory environments. Supporting both hand-drawn molecular structures and computer-generated diagrams, the system accommodates various input formats in-cluding PNG, JPEG, and BMP. This flexibility allows chemists to digitize legacy documents, process structures from publica-tions, or quickly translate their own sketches without requiring specialized chemical drawing software, thus streamlining the often tedious manual InChI generation process.

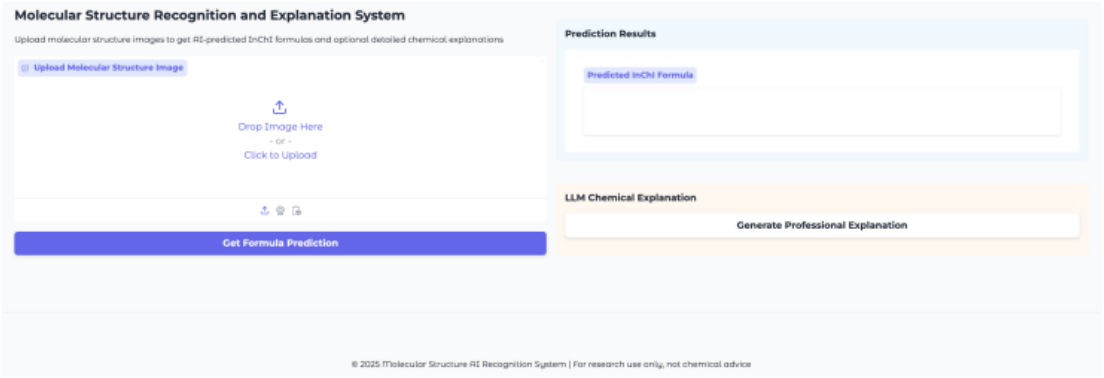

Figure 12. User Scenario 1

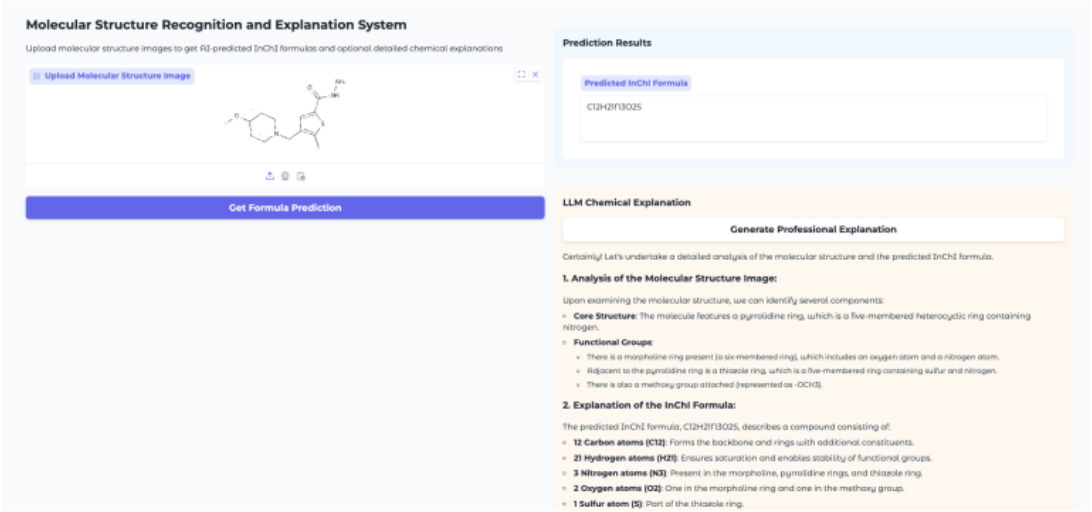

Figure 13. User Scenario 2

Once the user submits an image for analysis, the system processes it through our trained model and displays the pre-dicted InChI formula in a structured output area (Figure 13). For users seeking deeper insights, an optional "Generate Pro-fessional Explanation" button triggers the LLM component, which analyzes both the image and the predicted formula to provide a comprehensive assessment. This analyzes the image and formula to offer a detailed assessment, covering molecular composition, functional groups, potential applica-tions in pharmaceutical or industrial contexts, and, most im-portantly, validation of the accuracy of the prediction based on visual structure. This integration of neural machine translation with expert-level explanation capabilities creates a more com-plete solution for chemical documentation workflows, bridging the gap between visual representation and machine-readable chemical identifiers.

6. CONCLUSION

6.1. Key Contribution

This paper presents a systematic evaluation of configurable encoder-decoder architectures for automated InChI sequence generation from molecular images, which is a task crucial for digitizing chemical knowledge from diverse sources. Our key contribution lies in the comprehensive architectural analysis across 15 distinct configurations, revealing that ResNet-34 paired with GRU achieved the highest accuracy (82.97% on the 20K dataset) while EfficientNet-B0 with LSTM offered the optimal balance between performance (83.97% accuracy) and computational efficiency. We demonstrated that CNN-based encoders consistently outperform pure attention-based approaches like

Swin Transformer for molecular structure recognition in the small/medium size dataset, challenging assumptions about Transformer superiority in visual-linguistic tasks. Additionally, our teacher forcing strategy with gradual reduction improved model resilience during inference, and our attention mechanism enabled the model to focus on relevant molecular substructures during sequence generation. The re-sulting system achieved practical deployment in a web-based application that combines neural translation with contextual explanation capabilities, making chemical structure recogni-tion accessible to researchers without specialized software.

6.2. Future Work

Several promising directions could extend this research toward more robust and versatile molecular translation sys-tems. First, incorporating explicit chemical domain knowl-edge as constraints during sequence generation could sig-nificantly reduce chemically invalid predictions and improve stereochemical accuracy, particularly for complex structures with multiple chiral centers. Second, exploring multimodal training approaches that combine synthetic and hand-drawn molecular diagrams could enhance generalization to diverse image sources encountered in real-world applications. Third, investigating specialized visual encoders designed specifically for molecular graph representation could better capture the unique spatial relationships in chemical structures compared to general-purpose CNN architectures. Finally, extending the sys-tem to generate complete InChI strings beyond just molecular formulas would provide more comprehensive chemical iden-tification. This will require larger datasets and more sophis-ticated sequence modeling to handle the increased sequence length and complexity. These advancements would move the field closer to fully automated extraction of machine-readable chemical information from the vast repository of published structural diagrams in the scientific literature.

REFERENCES

[1] S. Wahler, T. M. Klapötke, and W. G. Proud, "Testing open-source tools for optical chemical structure recognition on novel nitrogen-rich energetic materials," Journal of Energetic Materials, 2023. [Online]. Available: https://doi.org/10.1080/07370652.2023.2215797

[2] N. M. O'Boyle, "Towards a universal smiles representation - a standard method to generate canonical smiles based on the inchi," Journal of Cheminformatics, vol. 4, no. 1, p. 22, Sep. 2012. [Online]. Available: https://doi.org/10.1186/1758-2946-4-22

[3] S. Heller, A. McNaught, S. Stein, D. Tchekhovskoi, and I. Pletnev, "Inchi-the worldwide chemical structure identifier standard," Journal of Cheminformatics, vol. 5, Jan. 2013. [Online]. Available: https://doi.org/10.1186/1758-2946-5-7

[4] A. P. Bento et al., "An open source chemical structure curation pipeline using rdkit," Journal of Cheminformatics, vol. 12, no. 1, sep. 2020. [Online]. Available: https://doi.org/10.1186/s13321-020-00456-1

[5] D. Jin et al., "Application of transformers to chemical synthesis," Molecules, vol. 30, no. 3, pp. 493–493, Jan. 2025. [Online]. Available: https://doi.org/10.3390/molecules30030493

[6] I. V. Filippov and M. C. Nicklaus, "Optical structure recognition software to recover chemical information: Osra, an open source solution," Journal of Chemical Information and Modeling, vol. 49, pp. 740–743, Feb. 2009. [Online]. Available: https://doi.org/10.1021/ ci800067r

[7] J. R. McDaniel and J. R. Balmuth, "Kekule: Ocr-optical chemical (structure) recognition," Journal of Chemical Information and Computer Sciences, vol. 32, pp. 373–378, Jul. 1992. [Online]. Available: https://doi.org/10.1021/ci00008a018

[8] A. T. Valko and A. P. Johnson, "Clide pro: The latest generation of clide, a tool for optical chemical structure recognition," Journal of Chemical Information and Modeling, vol. 49, pp. 780–787, Mar. 2009. [Online]. Available: https://doi.org/10.1021/ci800449t

[9] J. Staker, K. Marshall, R. Abel, and C. M. McQuaw, "Molecular structure extraction from documents using deep learning," Journal of Chemical Information and Modeling, vol. 59, pp. 1017–1029, Feb. 2019. [Online]. Available: https://doi.org/10.1021/acs.jcim.8b00669

[10] K. Rajan, A. Zielesny, and C. Steinbeck, "Decimer 1.0: deep learning for chemical image recognition using transformers," Journal of Cheminformatics, vol. 13, Aug. 2021. [Online]. Available: https://doi.org/10.1186/s13321-021-00538-8

[11] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature, vol. 521, no. 7553, pp. 436–444, may 2015. [Online]. Available: https://doi.org/10.1038/nature14539

[12] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," Communications of the ACM, vol. 60, no. 6, pp. 84–90, jun. 2017. [Online]. Available: https://doi.org/10.1145/3065386

[13] M. Oldenhof, Ád´am Arany, Y. Moreau, and J. Simm, "ChemGrapher: Optical Graph Recognition of Chemical Compounds by Deep Learning," Journal of Chemical Information and Modeling, vol. 60, no. 11, pp. 4731–4741, 2020. [Online]. Available: https://doi.org/10.1021/acs.jcim.0c00459

[14] H. Weir, K. Thompson, A. Woodward, B. Choi, A. Braun, and T. J. Martínez, "Chempix: Automated recognition of hand-drawn hydrocarbon structures using deep learning," Chemical Science, vol. 12, no. 31, pp. 10 622–10 633, aug. 2021. [Online]. Available: https://doi.org/10.1039/D1SC02957F

[15] I. Khokhlov, L. Krasnov, M. V. Fedorov, and S. Sosnin, "Image2SMILES: Transformer-Based Molecular Optical Recognition Engine," Chemistry–Methods, vol. 2, p. e202100069, 2022. [Online]. Available: https://doi.org/10.1002/cmtd.202100069

[16] J. Yi, C. Wu, X. Zhang, X. Xiao, Y. Qiu, W. Zhao, T. Hou, and D. Cao, "MICER: a pre-trained encoder–decoder architecture for molecular image captioning," Bioinformatics, 2022. [Online]. Available: https://doi.org/10.1093/bioinformatics/btac545

[17] Y. Li, G. Chen, and X. Li, "Automated Recognition of Chemical Molecule Images Based on an Improved TNT Model," Applied Sciences, vol. 12, no. 2, p. 680, 2022. [Online]. Available: https://doi.org/10.3390/app12020680

[18] Z. Xu, J. Li, Z. Yang, S. Li, and H. Li, "SwinOCSR: end-to-end optical chemical structure recognition using a Swin Transformer," Journal of Cheminformatics, vol. 14, p. 16, 2022. [Online]. Available: https://doi.org/10.1186/s13321-022-00624-5

[19] K. Rajan, H. O. Brinkhaus, M. I. Agea, A. Zielesny, and C. Steinbeck, "DECIMER.ai: an open platform for automated optical chemical structure identification, segmentation and recognition in scientific publications," Nature Communications, 2023. [Online]. Available: https://doi.org/10.1038/s41467-023-40782-0

[20] Y. Chen, C. T. Leung, Y. Huang, J. Sun, H. Chen, and H. Gao, "Molnextr: a generalized deep learning model for molecular image recognition," Journal of Cheminformatics, vol. 16, Dec. 2024. [Online]. Available: https://doi.org/10.1186/s13321-024-00926-w

[21] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), pp. 10 012–10 022, oct. 2021. [Online]. Available: https://doi.org/10.1109/ICCV48922.2021.00989

[22] H. O. Brinkhaus, K. Rajan, A. Zielesny, and C. Steinbeck, "Randepict: Random chemical structure depiction generator," Journal of Cheminformatics, vol. 14, Jun. 2022. [Online]. Available: https: //doi.org/10.1186/s13321-022-00609-4

[23] X. Fang, J. Wang, X. Cai, S. Chen, S. Yang, L. Yao, L. Zhang, and G. Ke, "Molparser: End-to-end visual recognition of molecule structures in the wild," arXiv preprint arXiv:2411.11098, Nov. 2024. [Online]. Available: https://doi.org/10.48550/arxiv.2411.11098

[24] Y. Qian, J. Guo, Z. Tu, Z. Li, C. W. Coley, and R. Barzilay, "MolScribe: Robust Molecular Structure Recognition with Image-to-Graph Generation," Journal of Chemical Information and Modeling, Mar. 2023. [Online]. Available: https://doi.org/10.1021/acs.jcim.2c01480

[25] Y. Wang, R. Zhang, S. Zhang, L. Guo, Q. Zhou, B. Zhao, X. Mo, Q. Yang, Y. Huang, K. Li, Y. Fan, L. Huang, and F. Zhou, "OCMR: A comprehensive framework for optical chemical molecular recognition," Computers in Biology and Medicine, vol. 161, p. 107187, 2023. [Online]. Available: https://doi.org/10.1016/j.compbiomed.2023.107187

[26] Information on: https://www.kaggle.com/competitions/bms-molecular-translation