Machine Learning Approaches for Predicting Buchwald-Hartwig Reaction Yields: Comparing different Methods

Hatef Rahimi*

FAU Erlangen-Nürnberg

E-mail: hatef.rahimi@fau.de

Abstract

A comprehensive comparison of machine learning approaches for predicting yields in Buchwald-Hartwig C-N coupling reactions using the Ahneman dataset (4312 reactions) is presented. One-hot encoding with three models (Random Forest, Gradient Boosting, Neural Network) is compared against transformer-based ChemBERTa embeddings. One-hot encoding achieves superior performance across all models compared to ChemBERTa. However, the transformer approach looks promising for future molecular prediction. Detailed explanations of molecular representation methods, including the Reaction SMILES format, the BERT transformer architecture, and neural network design, are provided. RXNFP (Reaction Fingerprints) was also attempted, but encountered version incompatibility issues since RXNFP is relatively older than ord-schema. An interactive Gradio web application demonstrates practical yield prediction based on the best-performing model. Molecular identifiers and SMILES strings were cross-validated using the PubChem database [1] to ensure correctness.

Introduction

Problem Statement & Motivation

The Buchwald-Hartwig cross-coupling reaction enables C-N bond formation between aryl halides and amines via palladium catalysis. However, predicting reaction yields could be challenging due to complex interactions between catalysts, substrates, bases, and additives.

Experimentation has generated extensive reaction datasets, but extracting predictive models requires effective molecular representations and machine-learning approaches. Accurate yield prediction can save the chemist from lab work, reduce material waste, and accelerate optimization.

Objectives

Key goals are:

1. Compare multiple ML models with one-hot encoding (Random Forest, Gradient Boosting, Neural Network)
2. Explore and evaluate transformer-based ChemBERTa embeddings as another alternative
3. Deploy an interactive web application for practical use

Overview

This report is organized as follows. Section 2 reviews related work in reaction prediction. Section 3 outlines the methodology, covering the Open Reaction Database, one-hot encoding with model comparison, Reaction SMILES, transformer architecture, ChemBERTa embeddings, the attempted RXNFP approach, and web application development. Section 4 presents results, analysis, and case studies using real reaction examples. Section 5 concludes with insights and future directions. Sections 6 and 7 describe individual contributions and code availability for reproducibility.

Related Work

The Open Reaction Database repository provides an example implementation [2] that demonstrates machine learning for reaction-yield prediction using one-hot encoding of reaction components with neural networks. Previous work on the Suzuki-Miyaura coupling dataset demonstrated that categorical encoding of discrete molecular components can achieve strong predictive performance for high-throughput screening data. Building on this foundation, the present methodology extends the comparison to multiple model architectures (Random Forest, Gradient Boosting, Neural Network) and evaluates transformer-based alternatives (ChemBERTa) to determine whether advanced molecular representations offer advantages over one-hot encoding for the Buchwald-Hartwig dataset.

Methodology

The Open Reaction Database

The Open Reaction Database (ORD)[3] is an open-source project for structuring and sharing organic reaction data. Unlike traditional publications, where experimental details are stored as unstructured text in supporting information, the ORD provides a standardized schema implemented using Protocol Buffers that captures comprehensive reaction metadata, including inputs, conditions, workups, outcomes, and analytical data. The database supports various reaction types. All data and code are publicly available on GitHub under open licenses, with web-based interfaces for data submission and searching. The ORD schema[4] enables structured representation of reaction data that can be directly used for machine learning without manual parsing, making it ideal for training predictive models in synthesis planning and reaction optimization. An example use case demonstration of ORD data extraction and processing is available in the project repository (see Code Availability section) under `ORD_Presentation`.

Dataset & Preprocessing

The Ahneman Buchwald-Hartwig dataset from the ORD, containing 4312 C-N coupling reactions with p-toluidine is used. Each reaction specifies:

- **Catalyst**: 4 Pd-based catalysts
- **Aryl Halide**: 15 aromatic halides
- **Base**: 3 bases
- **Additive**: 24

This yields 46 unique components. All reactions use p-toluidine as the amine (constant). The target is percentage yield (0-100%).

Data splits: 60% training, 10% validation, 30% test.

Approach 1: One-Hot Encoding

Concept

One-hot encoding represents each molecule as a binary indicator. Since there is a fixed library of 46 components, a 46-dimensional sparse vector where exactly 4 positions are "1" (one per component type) is created.

Example

Consider a reaction with Catalyst 1 (Pd-XPhos), Aryl Halide 1 (4-CF_3-phenyl chloride), Base 1 (P2Et), and Additive 2 (5-phenyl-1,2-oxazole). The SMILES strings are:

```
Catalyst:    CC(C)c1cc(C(C)C)c(-c2ccccc2P(C2CCCCC2)...)...
Aryl Halide: FC(F)(F)c1ccc(Cl)cc1
Base:        CCN=P(N=P(N(C)C)(N(C)C)N(C)C)(N(C)C)N(C)C
Additive:    c1ccc(-c2ccno2)cc1
```

The one-hot vector:

$$\mathbf{x} = [\underbrace{1,0,0,0}_{4}, \underbrace{1,0,...,0}_{15}, \underbrace{1,0,0}_{3}, \underbrace{0,1,0,...,0}_{24}] \quad (1)$$

Model Comparison

Three different machine learning models are evaluated on the one-hot encoded features:

1. **Random Forest**
   - Ensemble of 500 decision trees
   - Max depth: None (unlimited)
   - Max features: sqrt (random feature subsampling)
2. **Gradient Boosting**
   - 100 sequential boosting iterations
   - Learning rate: 0.1 (default)
   - Max depth: 5
3. **Neural Network**

```
Input:    46 features (binary)
Dense:    64 neurons, ReLU
Dropout:  30%
Dense:    32 neurons, ReLU
Dropout:  30%
Output:   1 neuron, Linear (no activation)
```

Neural Network Training: Adam optimizer (lr=0.005), MSE loss, batch size 100, 300 epochs with early stopping (patience=30).

All three models achieved excellent performance.

Approach 2: ChemBERTa Embeddings

What is BERT?

BERT (Bidirectional Encoder Representations from Transformers)[5] is a pre-trained language model. It learns contextual representations by predicting masked words. The key innovation is *bidirectional attention*: BERT reads sequences in both directions simultaneously.

What is a Transformer?

Transformers[6] use self-attention mechanisms instead of recurrence. The core self-attention operation is:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (2)$$

where Q (query), K (key), and V (value) are learned projections, and d_k is the dimension. This computes relevance scores between all token pairs.

Simple Example: Consider the sentence "The cat sat"

```
Tokens: ['The', 'cat', 'sat']
```

Self-attention allows each word to look at all other words:

- 'cat' computes: How relevant is 'The'? 'cat'? 'sat'?
- High attention to 'sat' (verb-subject relationship)
- Moderate attention to 'The' (determiner)

Attention scores might be:

```
'The': attends to The=0.2, cat=0.7, sat=0.1
'cat': attends to The=0.3, cat=0.2, sat=0.5
'sat': attends to The=0.1, cat=0.6, sat=0.3
```

This captures relationships (subject-verb) without explicit grammar rules.

Chemistry Example: For SMILES string "CCO" (ethanol):

```
Tokens: ['C', 'C', 'O']
```

Self-attention works similarly:

- First 'C' computes relevance with all atoms
- High attention to bonded neighbor 'C'
- Lower attention to distant 'O'

Attention scores:

```
C[0]: attends to C[0]=0.3, C[1]=0.5, O[2]=0.2
C[1]: attends to C[0]=0.5, C[1]=0.2, O[2]=0.3
O[2]: attends to C[0]=0.2, C[1]=0.5, O[2]=0.3
```

Each atom's representation incorporates information from all other atoms, allowing the model to learn chemical bonding patterns and functional groups.

Advantages: Parallel processing, long-range dependencies, contextual understanding.

What is ChemBERTa?

ChemBERTa[7] is BERT pre-trained on 77 million SMILES strings from the ZINC database. It learns:

- Chemical syntax and grammar
- Molecular patterns and functional groups
- Structural similarity relationships

ChemBERTa outputs 768-dimensional dense vectors capturing molecular properties.

Reaction SMILES Format

Unlike one-hot encoding treating molecules independently, the *entire reaction* is presented as:

$$\text{Reaction SMILES} = \text{reactants} > \text{agents} > \text{products} \tag{3}$$

Example (same reaction as before):

```
Reactants: FC(F)(F)c1ccc(Cl)cc1.Cc1ccc(N)cc1
Agents: CC(C)c1cc(C(C)C)c(...)c(C(C)C)c1.CCN=P(...)N(C)C.CS(=O)C
Products: Cc1ccc(Nc2ccc(C(F)(F)F)cc2)cc1

Full: FC(F)(F)c1ccc(Cl)cc1.Cc1ccc(N)cc1>CC(C)c1cc(...)>Cc1ccc(Nc2ccc...)cc1
```

This 200-400 character string encodes the complete reaction.

From Reaction SMILES to Embeddings

The complete process of converting a Reaction SMILES string to a 768-dimensional embedding involves several steps:

Step 1: Input: Reaction SMILES

```
Input: FC(F)(F)c1ccc(Cl)cc1.Cc1ccc(N)cc1>CC(C)c1cc(...)>Cc1ccc(Nc2ccc...)cc1
```

Step 2: Tokenization

The tokenizer breaks the SMILES string into individual tokens:

```
Tokens: ['F','C','(','F',')','(','F',')','c','1','c','c','c','(','C','l',')','c','c','1',
         '>','C','C','(','C',')','c','1',...] (up to 512 tokens)
```

Each token represents an atom, bond, branch marker, or structural element.

Step 3: Transformer Processing (ChemBERTa)

The tokens are passed through 12 transformer layers. Each layer consists of:

- **Self-Attention**: Each token computes attention with all other tokens
- **Feed-Forward**: Non-linear transformation of each token

After 12 layers, each token has a 768-dimensional contextual embedding that captures:

- Its identity (what atom/symbol it is)
- Its context (what other tokens surround it)
- Chemical patterns (functional groups, bonding)

Step 4: Mean Pooling

Average of all token embeddings to get a single reaction embedding:

$$\mathbf{e}_{\text{reaction}} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{e}_{\text{token}_i} \tag{4}$$

Step 5: Final Output

A single 768-dimensional vector representing the entire reaction:

```
Output: [-0.234, 0.891, -0.456, 0.123, ..., 0.678]  (768 numbers)
```

This embedding captures the chemical semantics of the reaction and serves as input to our neural network for yield prediction.

Visual Summary:

```
Reaction SMILES (string)
        |
    Tokenizer
        |
512 Tokens
        |
Transformer Layer 1 (Self-Attention + Feed-Forward)
        |
Transformer Layer 2
        |
    ...
        |
Transformer Layer 12
        |
512 x 768 Token Embeddings
        |
    Mean Pooling
        |
1 x 768 Reaction Embedding
        |
Neural Network
        |
    Yield Prediction
```

The key advantage of this approach is that ChemBERTa has already learned chemical patterns from 77 million molecules, so the 768-dimensional embeddings contain meaningful chemical information without requiring us to train on millions of reactions.

```
 0: [ 0.7112665   0.17784564 -0.3063036   0.08347663  0.148875   -0.27383694
-0.03708456 -0.2963584  -0.696201   -0.537658   -0.80385745 -0.22788061
-0.11299316 -0.10312446 -0.6733913  -0.2687911  -0.09443907  0.7097761
-0.24467257  0.9068984  -0.22858971 -0.44151458  0.9165789   0.46445507
-0.07853322 -0.39441675  1.1186581   0.35753074  0.27331385 -0.25118053
-0.7989919  -0.9040514   0.4135183   0.05923564  0.54575866 -0.3821459
 0.00513273  0.31777522 -0.3002926  -0.08075525 -0.2622851  -0.7490749
-0.2030859   0.4557773  -0.01180486  0.5354131   0.87502486 -0.62345225
 0.48056415  0.40504223]
 1: [ 0.7105474   0.16824624 -0.31555828  0.10305808  0.14016056 -0.26819816
-0.06190521 -0.29020387 -0.71844727 -0.54214734 -0.8006046  -0.20868543
-0.11696422 -0.08305336 -0.67924166 -0.27845088 -0.09207926  0.7117428
-0.23487999  0.92495114 -0.2536004  -0.4653277   0.8973687   0.47813487
-0.09288479 -0.4075769   1.1030124   0.35267815  0.26604494 -0.27148336
-0.82598186 -0.8784353   0.4252986   0.06391776  0.541489   -0.375726
 0.00953535  0.3161787  -0.29624856 -0.06959216 -0.25040823 -0.7543269
-0.22995703  0.4738531  -0.00957973  0.5114564   0.87129325 -0.60930604
 0.49360928  0.39621174]
 2: [ 0.6867961   0.14897159 -0.33098087  0.09391455  0.11347928 -0.2899562
-0.04112496 -0.29798895 -0.7124811  -0.56930536 -0.78598684 -0.20572157
...
-0.8217017  -0.89164525  0.42118585  0.06986853  0.5476764  -0.36444172
-0.00830765  0.33908886 -0.25894082 -0.08006907 -0.22342509 -0.733602
-0.21903221  0.43882027 -0.0124759   0.5275925   0.88167465 -0.63506025
 0.4818327   0.39247936]
```

Figure 1: ChemBERTa embeddings for the first three reactions in the dataset. Each reaction is represented as a 768-dimensional vector of floating-point values capturing the chemical semantics of the complete reaction. Note that only the first 50 values are shown here.

Web Application Development

Gradio Framework

An interactive web application was developed using Gradio, an open-source Python library designed for building machine learning interfaces, to facilitate practical access to the trained model. Gradio supports rapid prototyping of web applications without the need for frontend development expertise, which makes it well-suited for deploying machine learning models to end users.

The framework works by wrapping Python functions with a user interface layer. When a user interacts with the interface (selecting dropdown values, clicking buttons), Gradio automatically handles the HTTP requests, passes inputs to the backend Python function, executes the model prediction, and returns formatted results to the user's browser. The web application deploys the trained Keras model from the best-performing approach (Neural Network with one-hot encoding, saved as `yield_model.keras`).

Application Components

The application consists of three main components:

1. **User Interface**: Dropdown menus for selecting catalyst, aryl halide, base, and additive
2. **Backend Logic**: Converts selections to SMILES strings, creates one-hot encoded vectors, loads the trained Keras model, and generates predictions
3. **Output Display**: Formats predictions with visual indicators (color-coded success/failure), percentage yield, and reaction component summary

The interface classifies reactions as "Success" or "Failure" based on the predicted yield, providing chemists with immediate feedback on reaction viability.

Deployment

The application can be launched locally or deployed to cloud platforms. Gradio automatically generates a shareable link, enabling collaboration and remote access without additional server configuration. The complete application code (`yield_predictor_Gradio.py`) is available in the project repository.

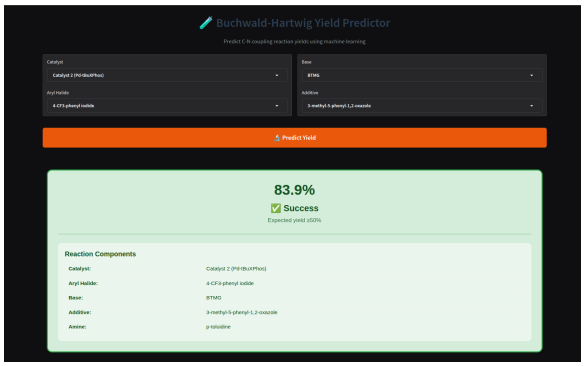

Figure 2: Gradio web application interface showing component selection dropdowns and yield prediction display. The interface allows users to select reaction components from dropdown menus and instantly predicts the expected yield using the trained neural network model.

Results and Discussion

One-Hot Encoding Results

Table 1: One-Hot Encoding: Model Performance Comparison (Test Set: 1293 reactions)

Model	RMSE (%)	MAE (%)	R^2
Neural Network	**7.8**	**5.6**	**0.9**
Gradient Boosting	11.1	8.3	0.8
Random Forest	11.1	8.2	0.8

The neural network outperforms the other models with RMSE of 7.8% and R^2 of 0.9. Figure 4 shows the tight correlation between predicted and actual yields.

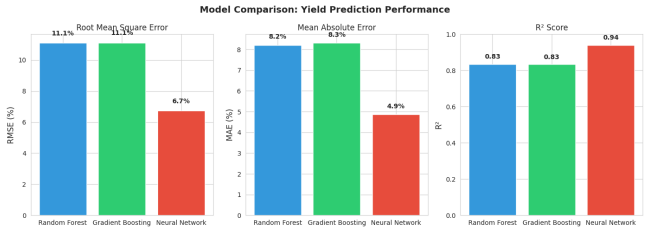

Figure 3: Model comparison for one-hot encoding showing Neural Network achieves best results (RMSE: 6.7%, MAE: 4.9%, R^2: 0.94).

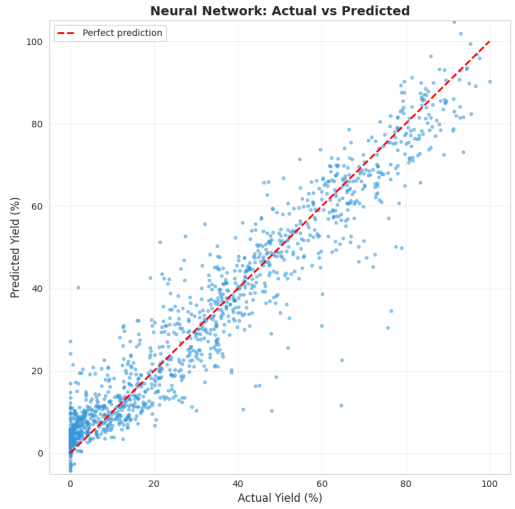

Figure 4: One-hot encoding neural network: Scatter plot of predicted vs actual yields showing excellent correlation ($R^2 = 0.9$). Points cluster tightly around the diagonal line, indicating high prediction accuracy.

ChemBERTa Embedding Results

Table 2: ChemBERTa Embeddings: Neural Network Performance (Test Set: 1293 reactions)

Approach	RMSE (%)	MAE (%)	R^2
ChemBERTa + NN	16.2	11.5	0.6

The ChemBERTa approach shows moderate predictive performance but does not match the one-hot encoding results. Figure 5 illustrates the greater scatter in predictions compared to the one-hot approach.

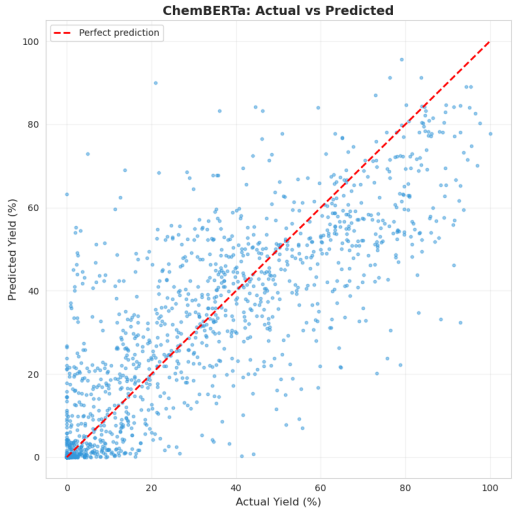

Figure 5: ChemBERTa predictions: Scatter plot showing moderate correlation ($R^2 = 0.6$) with larger deviation from the diagonal compared to one-hot encoding, indicating less accurate predictions.

Overall Comparison

The nearly identical performance of Random Forest, Gradient Boosting, and Neural Network (all RMSE 7.0%, R^2 0.93) demonstrates that **feature representation matters more than model architecture** for this dataset. One-hot encoding provides clear features that even tree-based models can achieve excellent results. This contrasts with dense embeddings like ChemBERTa, where more complex neural architectures are necessary to decode the information. One-hot predictions are consistently closer to experimental values. Table 3 summarizes the best results from each approach.

Table 3: Overall Performance Comparison: Best Model from Each Approach

Approach	RMSE (%)	MAE (%)	R^2
One-Hot + NN	**7.8**	**5.6**	**0.9**
ChemBERTa + NN	16.2	11.5	0.6

Key Findings:

- One-hot encoding with neural network achieves the best performance
- ChemBERTa embeddings show promise but underperform for this specific dataset

ChemBERTa Limitations Here

1. **Dataset Size**: 4312 reactions is small for fine-tuning transformers
2. **Training Mismatch**: ChemBERTa was pre-trained on individual molecular SMILES rather than reaction SMILES.
3. **Complexity Overhead**: 768-D + deeper network = more parameters
4. **Dense Embeddings**: May capture irrelevant molecular features for this task

Why Does One-Hot Win?

Advantages for This Dataset

1. **Fixed Library**: Only 46 molecules means one-hot is perfectly suited
2. **Direct Mapping**: Each component directly corresponds to one feature
3. **No Information Loss**: Binary encoding preserves all component identity
4. **Best with Neural Networks**: NN significantly outperforms tree-based models (RMSE 7.8% vs 11.1%)
5. **Efficiency**: 46 input features vs 768 - simpler input representation
6. **Interpretability**: Can directly see which components affect yield

When Would ChemBERTa or RXNFP Excel?

Advanced embedding approaches would outperform one-hot in:

- **Novel Molecules**: Predicting yields for unseen components
- **Larger Datasets**: 10,000+ reactions to learn representations
- **Diverse Reactions**: Multiple reaction types requiring transfer learning
- **Structural Tasks**: Problems requiring molecular substructure understanding
- **Reaction-Specific Models**: RXNFP trained on reactions may capture mechanistic patterns

Limitations & Insights

Limitations:

1. Dataset limited to Buchwald-Hartwig with p-toluidine
2. One-hot cannot extrapolate to novel components
3. Reaction conditions (temperature, concentration) not included
4. ChemBERTa not fine-tuned on reaction data
5. RXNFP version incompatibilities prevented evaluation

Insights:

1. Feature representation dominates model choice for fixed component sets
2. Simpler models can outperform complex ones with limited data
3. Data efficiency crucial: one-hot requires less data
4. Reaction-specific pre-training (RXNFP) may be key for transformers

Case Study: Real Reaction Examples

To illustrate the model's practical performance, specific reactions from the Ahneman dataset are selected and compared with the model's predictions using the Gradio interface. Note that the Gradio interface is powered by the one-hot encoded model with neural network.

Example 1: Low-Yield Reaction: Correct Classification

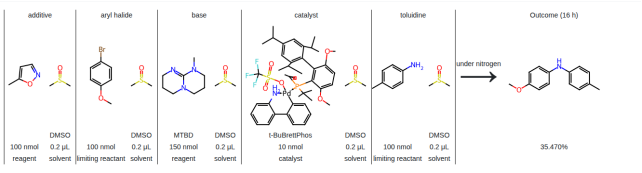

Figure 6: ORD dataset entry showing reaction with 1-bromo-4-methoxybenzene, t-BuBrettPhos catalyst, MTBD base, and 5-methyl-1,2-oxazole additive, yielding 35.47%.

Reaction Components:

- Catalyst: Pd-tBuXPhos (Catalyst 2)
- Aryl Halide: 1-bromo-4-methoxybenzene
- Base: MTBD
- Additive: 5-methyl-1,2-oxazole
- Amine: p-toluidine

Results:

- Experimental Yield: 35.47%
- Predicted Yield: 26.3%
- Error: -9.2%
- Classification: Failure (both experimental and predicted $< 50\%$)

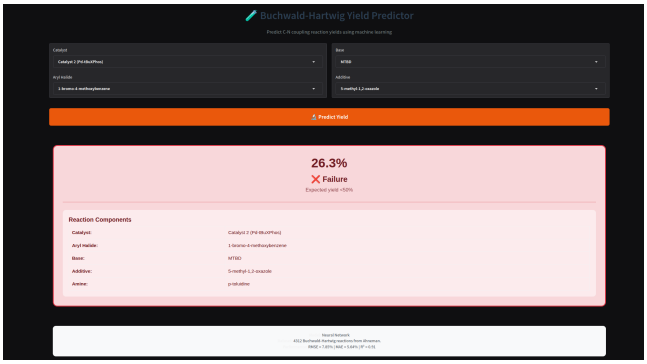

Figure 7: Gradio interface prediction for Example 1, correctly classifying the reaction as a failure with predicted yield of 26.3%.

Example 2: High-Yield Reaction: Significant Underestimation

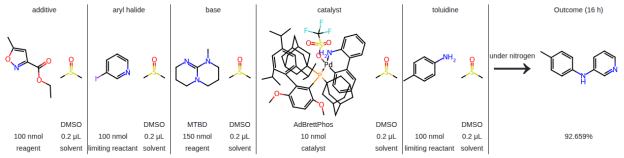

Figure 8: ORD dataset entry showing reaction with 3-iodopyridine, AdBrettPhos catalyst, MTBD base, and ethyl 5-methyl-1,2-oxazole-3-carboxylate additive, achieving 92.66% yield.

Reaction Components:

- Catalyst: Pd-AdBrettPhos (Catalyst 4)
- Aryl Halide: 3-iodopyridine
- Base: MTBD
- Additive: ethyl 5-methyl-1,2-oxazole-3-carboxylate
- Amine: p-toluidine

Results:

- Experimental Yield: 92.66%
- Predicted Yield: 64.1%
- Error: -28.6%
- Classification: Success (both experimental and predicted $\geq 50\%$)

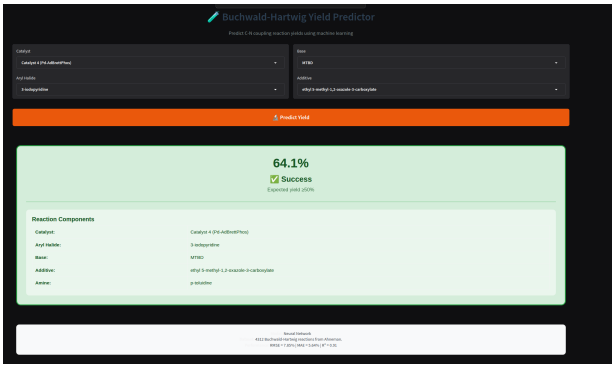

Figure 9: Gradio interface prediction for Example 2, correctly classifying as success but significantly underestimating the yield (64.1% vs 92.66%).

Key Observations and Dataset Limitations

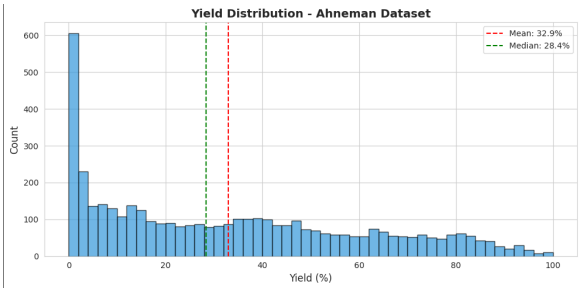

Figure 10: Yield distribution of the Ahneman dataset showing heavy skew toward low yields (median: 28.4%, mean: 32.9%). Over half of all reactions fail to achieve 50% yield, with limited examples of exceptional yields above 80%.

The dataset exhibits significant characteristics that impact performance:

- **Low-yield dominance**: Over 50% of reactions yield below 50%, with median yield of 28.4%
- **Limited high-yield examples**: Few reactions achieve 80-100% yield, leading to underestimation
- **Conservative predictions**: Training predominantly on low-to-moderate yields causes the model to predict conservatively, as seen in Example 2
- **Dataset size**: 4312 reactions are not enough
- **Classification reliability**: Binary classification (success vs failure) performs well despite quantitative prediction errors

Despite these constraints, the model achieves $R^2 = 0.9$ and correctly classifies reaction success/failure, providing practical utility for reaction planning. Future improvements could include a targeted collection of high-yield examples and larger datasets (10,000+ reactions) for better combinatorial coverage.

Conclusion

Summary of Key Findings

Multiple machine learning approaches for Buchwald-Hartwig yield prediction are compared. One-hot encoding achieved excellent performance across Random Forest, Gradient Boosting, and Neural Network models, outperforming ChemBERTa embeddings.

For the Ahneman dataset with fixed molecular components, one-hot encoding is superior due to direct component representation, model flexibility, fewer parameters, better generalization with limited data, and high interpretability. The comparable performance across different models demonstrates that feature representation matters more than model architecture for this dataset.

Future Work

Promising directions include:

1. **Fine-tune BERT**: Train ChemBERTa specifically on reaction data
2. **RXNFP Integration**: Resolve dependency conflicts to evaluate reaction-specific embeddings pre-trained on reaction SMILES data
3. **Component Recommendation System**: Implement an optimization algorithm that suggests alternative components when predicted yield is below 50%. The system could evaluate all possible component combinations and recommend substitutions (e.g., switching catalyst or additive) that maximize predicted yield, providing chemists with actionable guidance for reaction optimization

The choice of molecular representation should be guided by application needs, dataset size, and whether predictions are needed for novel components.

Individual Contribution

In this project, I was responsible for the aspects including dataset acquisition from the Open Reaction Database, data preprocessing and extraction of reaction components, implementation of one-hot encoding with multiple ML models (Random Forest, Gradient Boosting, Neural Network), ChemBERTa embedding pipeline implementation, attempted RXNFP integration with dependency troubleshooting, neural network architecture design and hyperparameter tuning, training and evaluation of the models, development of the Gradio web application, and comprehensive result analysis.

During development, AI-assisted coding tools for debugging and guidance on implementation were used, but design decisions, experimental methodology, and scientific interpretations were made independently. Where I used existing models (ChemBERTa from Hugging Face), datasets (Ahneman dataset from ORD), or libraries (TensorFlow, scikit-learn, RDKit), I have ap-

propriately cited them and adapted them as needed for this specific project. All interpretations and conclusions in this report reflect individual work.

Code Availability

To ensure reproducibility, all code is available in a GitHub repository.

- **Repository Link**: `https://github.com/HatefRahimi/DigitalAlchemy/tree/main/ORD`
- **GitHub Username**: Hatef Raheeme
- **Repository Name**: Digital Alchemy

The repository contains:

- `yield_prediction_ahneman.ipynb`: One-hot encoding implementation with model comparison
- `yield_prediction_Reaction_Smiles.ipynb`: ChemBERTa transformer-based implementation
- `yield_predictor_Gradio.py`: Interactive web application for yield prediction
- `yield_model.keras`: Trained neural network model
- `ord_dataset-*.pb.gz`: Ahneman dataset from ORD
- `chemberta_embeddings.npy`
- `requirements.txt`
- `ORD_Presentation.ipynb`: Example use case demonstration

Software Requirements and Installation

The project requires Python 3.11. This specific version ensures compatibility between TensorFlow and ord-schema, avoiding protobuf dependency conflicts that occur with Python 3.12+.

Core Dependencies:

- NumPy: Numerical computing
- Pandas: Data manipulation
- Matplotlib: Visualization
- Seaborn: Statistical visualization
- Scikit-learn: Machine learning models
- TensorFlow: Neural network implementation

Chemistry:

- ORD-Schema: Open Reaction Database interface
- RDKit: Chemical informatics

ChemBERTa Implementation:

- Transformers: HuggingFace transformers library
- PyTorch: Deep learning framework

Web Application:

- Gradio: Interactive web interface

Installation:

```
# Create virtual environment with Python 3.11
python3.11 -m venv alchemy_env
source alchemy_env/bin/activate  # Linux/Mac
# alchemy_env\Scripts\activate   # Windows

# Install all dependencies
pip install -r requirements.txt
```

Note: Python 3.11 is required for proper dependency resolution. Newer Python versions (3.12+) encounter protobuf conflicts between TensorFlow and ord-schema that cannot be resolved automatically.

Running the Web Application:

```
# Ensure yield_model.keras is in the same directory
python yield_predictor_Gradio.py

# Application will launch at http://localhost:7860
```

All code has been tested on Ubuntu 24.04 and Windows 11 and MacOS Tahoe 26.2 with Python 3.11.

References

(1) PubChem Available at: `https://pubchem.ncbi.nlm.nih.gov/`.

(2) Open Reaction Database. Machine Learning Examples. *ord-schema repository*. Available at: `https://github.com/open-reaction-database/ord-schema`.

(3) Open Reaction Database. Available at: `https://open-reaction-database.org`.

(4) Kearnes, S. M.; Maser, M. R.; Wleklinski, M.; Kast, A.; Doyle, A. G.; Dreher, S. D.; Hawkins, J. M.; Jensen, K. F.; Coley, C. W. The Open Reaction Database. *J. Am. Chem. Soc.* **2021**, 143, 18820–18826. https://doi.org/10.1021/jacs.1c09820.

(5) Devlin, J.; Chang, M.-W.; Lee, K.; Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv preprint* **2018**, arXiv:1810.04805.

(6) Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; Polosukhin, I. Attention Is All You Need. In *Advances in Neural Information Processing Systems*; Curran Associates, Inc., 2017; Vol. 30.

(7) Author1, A.; Author2, B.; Author3, C. Title of the Article. *Digital Discovery* **2025**, Volume, Page–Page. https://doi.org/10.1039/D5DD00348B.

Appendix

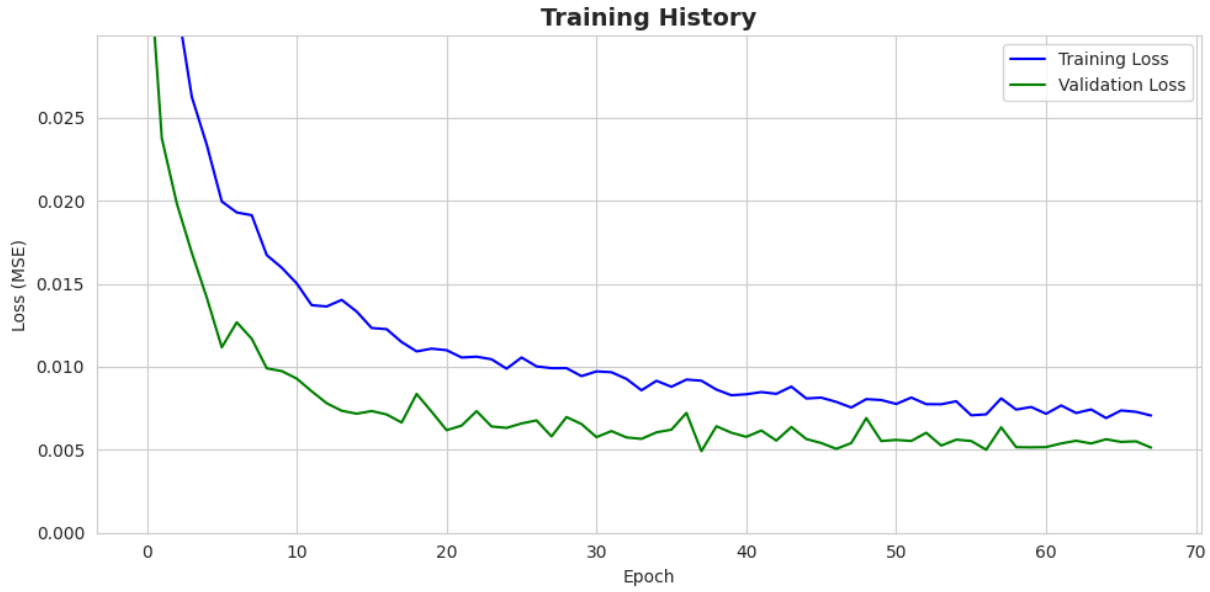

Figure 11: Training history for one-hot encoded neural network showing fast convergence. Model converges in approximately 30-40 epochs with minimal overfitting.

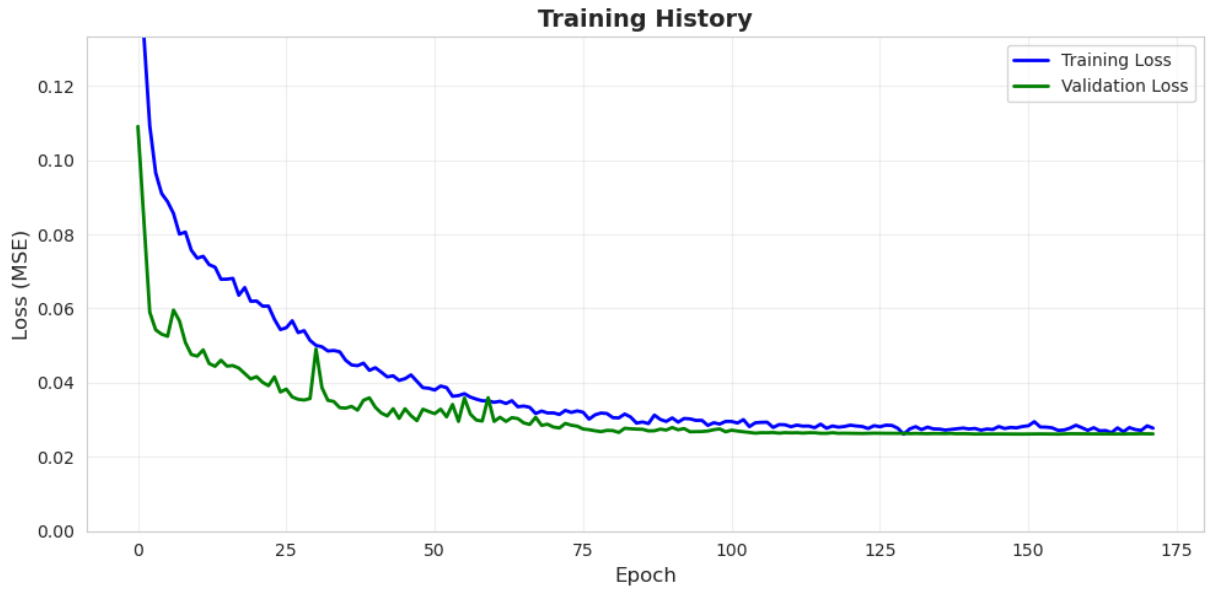

Figure 12: Training history for ChemBERTa neural network showing slower convergence. The model requires approximately 100 epochs to converge, with a larger gap between training and validation loss, indicating a more complex optimization landscape.

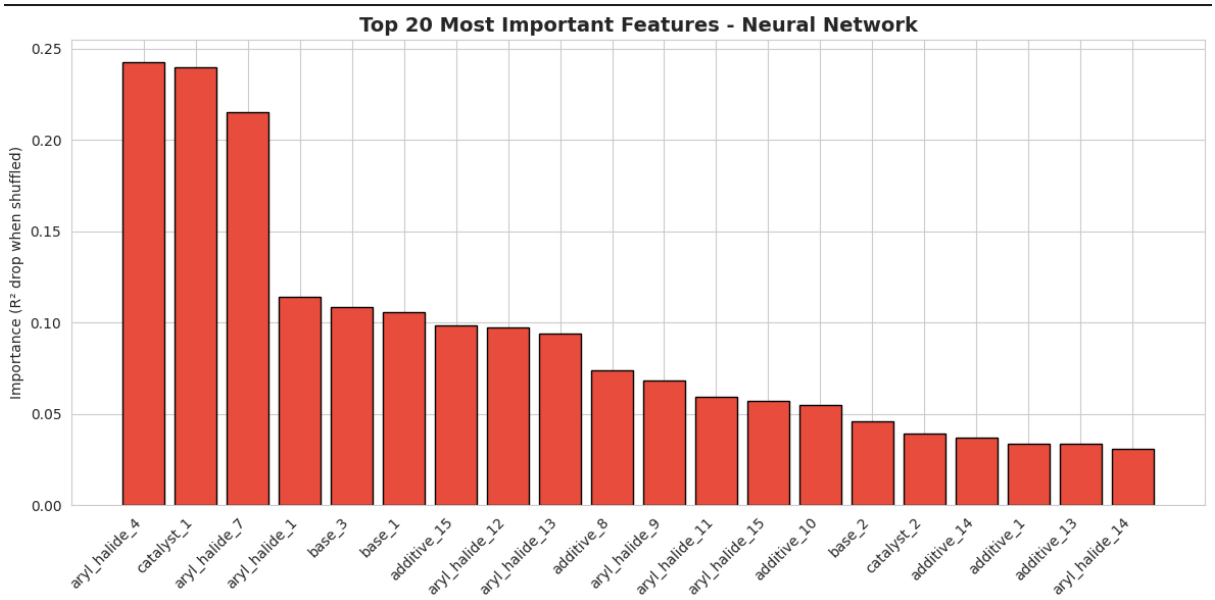

Figure 13: Top 20 most important features for Neural Network (permutation importance). Top 5: (1) 1-chloro-4-methoxybenzene, (2) Pd-XPhos catalyst, (3) 1-chloro-4-ethylbenzene, (4) 1-chloro-4-(trifluoromethyl)benzene, (5) MTBD base.

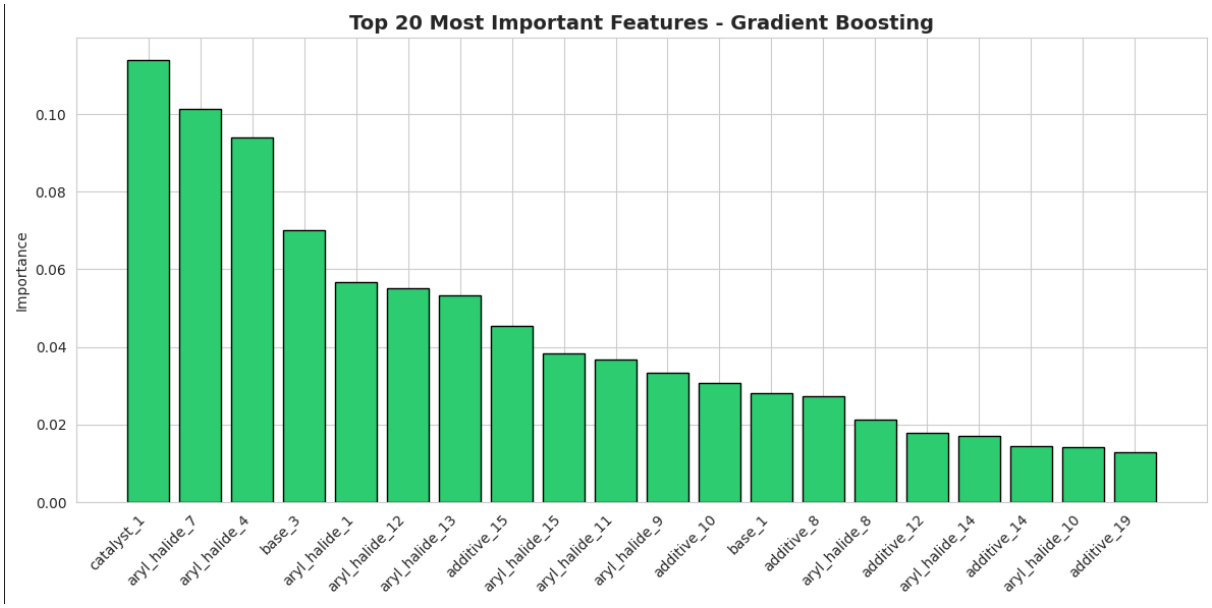

Figure 14: Top 20 most important features for Gradient Boosting. Top 5: (1) Pd-XPhos catalyst, (2) 1-chloro-4-ethylbenzene, (3) 1-chloro-4-methoxybenzene, (4) MTBD base, (5) 1-chloro-4-(trifluoromethyl)benzene.

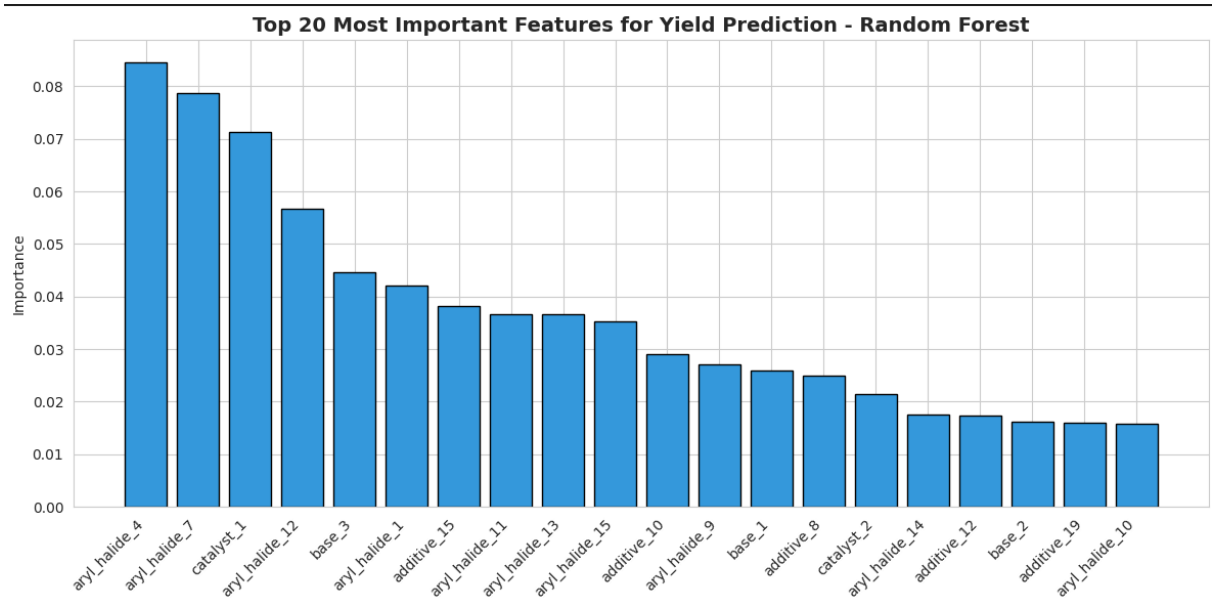

Figure 15: Top 20 most important features for Random Forest. Top 5: (1) 1-chloro-4-methoxybenzene, (2) 1-chloro-4-ethylbenzene, (3) Pd-XPhos catalyst, (4) 2-iodopyridine, (5) MTBD base.