# Supplementary Information

# EnZymClass: Substrate specificity prediction tool of plant Acyl-ACP Thioesterases based on Ensemble Learning

Deepro Banerjee1¶, Michael A. Jindra2¶, Alec J. Linot 2, Brian F. Pfleger2, Costas D. Maranas3\*

1 The Bioinformatics and Genomics Program, Huck Institutes of the Life Sciences, The Pennsylvania State University, University Park, Pennsylvania, United States of America

2 Department of Chemical and Biological Engineering, University of Wisconsin-Madison, Madison, Wisconsin, United States of America

3 Department of Chemical Engineering, The Pennsylvania State University, University Park, Pennsylvania, United States of America

\* Corresponding Author

E-mail: [cdm8@psu.edu](mailto:cdm8@psu.edu) (CDM)

¶ These authors contributed equally to this work

## Text S1: Description of the 47 feature extraction techniques used in EnZymClass

### Kernel methods

Three kernel based methods, described in this section, were used in EnZymClass. They were all implemented using the KeBABS software package [1].

#### Spectrum kernel

The k-spectrum kernel proposed by Leslie et. al. [2] is the set of all k-length contiguous subsequences that can occur in a given input sequence. If we assume to be the set of all characters of a sequence, to be a k-mer and to be function that denotes the number of times a given k-mer is present in a protein sequence , then the feature map is as follows:

#### Mismatch kernel

The mismatch kernel is a generalized form of the spectrum kernel. The kernel has two parameters k and m, where k is the length of all possible k-mers that might be present in a protein sequence and m is the maximum number of mismatches allowed for a single k-mer. If we assume to be the set of all characters of a sequence, to be a k-mer, then for a fixed k-mer u, the () mismatch pattern is the set of all -length subsequences generated from that differ from by at most mismatches. The feature map of a fixed k-mer u can be defined as:

Where is 1 if belongs to the set of -length subsequences that differ from by at most mismatches, 0 otherwise. The feature map on an input sequence is defined as:

#### 

#### Gappy pair kernel

The gappy pair kernel looks for pairs of matching -mers separated by a fixed number of amino acids in between them. It has two parameters and , where k denotes the length of the -mers and denotes the maximum gap between the matching amino acid -mer pairs to consider.

### N-gram methods

Two n-gram based methods, described below, were used in EnZymClass. They were all implemented using ngrampro package created as a part of this project. It can be accessed using pip, the python package manager. Accessibility and usage details of ngrampro [3] is available here: <https://pypi.org/project/ngrampro/>

#### k-mer motif builder

This representation is similar to the N-gram representation for language models [4]. Here, the entire training dataset (all enzyme sequences) is scanned to search for all possible sequence motifs of length k. A one hot encoded feature set is created with sequence motifs which are repeated in at least 2 instances of enzymes in the training data. This step is taken for two reasons, 1) To avoid building a large set of features and 2) It is expected that enzymes of a specific substrate specificity will have common motifs and the main aim of the model is to learn these common motifs, not a motif specific to a particular enzyme.

#### Grouped amino acid encoded k-mer motif builder

The grouped amino acid encoded k-mer motif builder performs an additional pre-processing step on the primary sequences of the enzymes before encoding them into a feature vector representation similar to the k-mer motif builder. In the preprocessing step, 20 amino acid types which can occur at a particular position in the sequence of an enzyme were categorized into five classes according to their physicochemical properties, hydrophobicity, charge, and molecular size. The five classes are the aliphatic group (: GAVLMI), aromatic group (: FYW), positive charge group (: KRH), negative charged group (: DE) and uncharged group (: STCPNQ). Finally, each amino acid in the sequence was replaced by its respective class to create an encoded sequence with only five different types of values per position. For example, a primary sequence ‘MLTPWS’ is represented as . Using the encoded representation, k-mer motif builder was used to represent the protein sequence. It should be noted that the preprocessing step significantly reduces the set of characters of the sequence and can help prevent overfitting since the kernel feature map size will be much lower than the original feature map size obtained by the k-mer motif builder. The reduced feature map size will require much lower number of parameters to train a classifier and thus may prevent overfitting.

### Physicochemical encodings

21 physicochemical encoding based methods, described below, were used in EnZymClass. They were all implemented using the ifeatpro [5] software package easily installable through pip (link: <https://pypi.org/project/ifeatpro/>). Ifeatpro is based on iFeature [6], a python tool to create various numerical descriptors of protein sequences.

#### Amino Acid Composition (AAC)

AAC encoding calculates the frequency of each of the 20 types of amino acids in a protein sequence and encodes them in a vector of length 20. Amino acid composition for a protein sequence x can be defined as:

Where is the count of an amino acid , is the length of the protein sequence and is the set of all amino acid types.

#### Composition of k-spaced Amino Acid Pairs (CKSAAP)

CKSAAP encoding calculates the frequency of amino acid pairs separated by at most k residues. For example, for k=2, the feature vector can be defined as:

Where can be any of the 20 different amino acid types and represents the total number of 2-spaced amino acid pairs present in a protein sequence.

#### Tri-Peptide Composition (TPC)

TPC calculates the frequency of the tripeptides in a protein sequence. The TPC encoding of a protein x can be represented as:

Where is the number of tripeptides represented by amino acid types , and that belongs to the set of all amino acids .

#### Di-Peptide Composition (DPC)

DPC calculates the frequency of the dipeptides in a protein sequence. The DPC encoding of a protein x can be represented as:

Where is the number of dipeptides represented by amino acid types and that belongs to the set of all amino acids .

#### Dipeptide Deviation from Expected Mean (DDE)

DDE feature vector is a function of three variables, dipeptide composition (), theoretical mean () and theoretical variance () used to calculate the final feature encoding. is defined in the Di-Peptide Composition section. is calculated as:

Where is the number of codons that code for amino acid , is the number of codons that code for amino acid and is the total number of possible codons excluding the stop codons. is calculated as:

Where is the length of the protein peptide. The is calculated as:

#### Grouped Amino Acid Composition (GAAC)

GAAC encoding performs an additional preprocessing step of categorizing the amino acids into five classes according to their physicochemical properties before computing the composition of the five groups similar to [AAC](#_Amino_Acid_Composition). The five classes are the aliphatic group (: GAVLMI), aromatic group (: FYW), positive charge group (: KRH), negative charged group (: DE) and uncharged group (: STCPNQ).

#### Composition of k-spaced Amino Acid Group Pairs (CKSAAGP)

The CKSAAGP encoding performs an additional preprocessing step of categorizing the amino acids into five classes according to their physicochemical properties before computing the frequency of the grouped amino acid pairs separated by at most residues similar to [CKSAAP](#_Composition_of_k-spaced). The five classes are the aliphatic group (: GAVLMI), aromatic group (: FYW), positive charge group (: KRH), negative charged group (: DE) and uncharged group (: STCPNQ).

#### Grouped Di-Peptide Composition (GDPC)

The GDPC encoding performs an additional preprocessing step of categorizing the amino acids into five classes according to their physicochemical properties before computing the dipeptide composition of the amino acid groups similar to [DPC](#_Di-Peptide_Composition_(DPC)). The five classes are the aliphatic group (: GAVLMI), aromatic group (: FYW), positive charge group (: KRH), negative charged group (: DE) and uncharged group (: STCPNQ).

#### Grouped Tri-Peptide Composition (GTPC)

The GTPC encoding performs an additional preprocessing step of categorizing the amino acids into five classes according to their physicochemical properties before computing the tripeptide composition of the amino acid groups similar to [TPC](#_Tri-Peptide_Composition_(TPC)). The five classes are the aliphatic group (: GAVLMI), aromatic group (: FYW), positive charge group (: KRH), negative charged group (: DE) and uncharged group (: STCPNQ).

#### Moran Correlation (Moran)

Moran descriptor is computed based on the numerical values representing the biological and physicochemical attributes of different types of amino acids given in the AAindex database [7]. The descriptor is defined as

Where is the lag in correlation, is the maximum value of the lag, and are the properties of amino acids at positions and as obtained from the AAindex database and is the average value of the property P across the entire protein sequence.

#### Geary Correlation (Geary)

Geary descriptor is computed based on the numerical values representing the biological and physicochemical attributes of different types of amino acids given in the AAindex database. The descriptor is defined as

Where is the lag in correlation, is the maximum value of the lag, and are the properties of amino acids at positions and as obtained from the AAindex database and is the average value of the property across the entire protein sequence.

#### Normalized Moreau-Broto Autocorrelation (NMBroto)

NMBroto descriptor is defined as

Where is the length of the protein sequence, is the lag in autocorrelation, is the maximum value of lag to be considered and is the Moreau-Broto autocorrelation descriptor calculated as:

Here, and are the properties of amino acids at positions and as obtained from the AAindex database.

#### Composition Transition Distribution – Composition (CTDC)

CTDC feature encoding transforms the amino acid sequence of a protein into three subgroups of 13 types of physicochemical properties including hydrophobicity, normalized Van der Waals Volume, polarity, polarizability, charge, secondary structures, and solvent accessibility. After dividing the sequence into subgroups of a specific physicochemical property, the frequencies of these subgroups are calculated, and the feature descriptor is described as follows:

Where denotes the subgroup type, denotes the set of all subgroups and denotes the physicochemical property.

#### Composition Transition Distribution – Transition (CTDT)

CTDC feature encoding transforms the amino acid sequence of a protein into three subgroups of 13 types of physicochemical properties including hydrophobicity, normalized Van der Waals volume, polarity, polarizability, charge, secondary structures, and solvent accessibility. After dividing the sequence into subgroups of a specific physicochemical property, the frequencies of the transition from one subgroup to another are calculated, and the feature descriptor is described as follows:

Where and denote the subgroup type, denotes the set of all subgroups and denotes the physicochemical property.

#### Composition Transition Distribution – Distribution (CTDD)

CTDD feature encoding transforms the amino acid sequence of a protein into three subgroups of 13 types of physicochemical properties including hydrophobicity, normalized Van der Waals volume, polarity, polarizability, charge, secondary structures, and solvent accessibility. After dividing the sequence into subgroups of a specific physicochemical property, the fraction of the entire sequence where the first residue, 25%, 50%, 75% and 100% of residues of any given subgroup is marked. The process is followed for all physicochemical properties and the numerical fractions are combined in a vector to represent the feature descriptor.

#### Conjoint Triad (CTriad)

CTriad considers three contiguous amino acids as a single unit. It creates a vector space where each element in the vector represents a triad type and another vector space that is the frequency vector corresponding to where each element denotes the number of triad type appearing in the protein sequence. Since the values of are directly correlated to the length of the protein, the numerical vector is normalized using Min-Max scaling technique.

#### k-spaced Conjoint Triad (KSCTriad)

KSCTriad not only takes into account contiguous amino acids to create the feature space as in CTriad but also considers amino acids that are separated by at most residues as a single unit.

#### Sequence-Order-Coupling Number (SOCNumber)

SOCNumber is calculated from the distance matrices proposed by Schneider-Wrede and Grantham. It is defined as:

Where is the entry in a distance matrix that denotes the distance between the two amino acids at position and , is the lag or the positional difference between two amino acid residues, is the maximum value of the lag to be considered and is the length of the protein sequence.

#### Quasi-sequence-order (QSOrder)

QSOrder is based on SOCNumber and is defined as:

Where is the normalized count of amino acid , is a weighing factor, is the SOCNumber, is an amino acid type and is the set of all amino acids. The above definition is valid for up to 20. For above 20, the definition is:

#### Pseudo-Amino Acid Composition (PAAC)

PAAC is defined as:

Where is the normalized count of amino acid , is a weighing factor and is a sequence order correlated factor defined as:

Where is the length of the protein sequence, is an integer parameter to be chosen that must be less than , and is defined as:

Where is the -th property in the amino acid property set for amino acid . The amino acid properties are normalized hydrophobicity values, normalized hydrophilicity values and normalized side chain masses of the 20 amino acids.

#### Amphiphilic Pseudo-Amino Acid Composition (APAAC)

APAAC is similar to PAAC and is defined as:

Where is the normalized count of amino acid , is a weighing factor and is a sequence order correlated factor defined as:

Where is the length of the protein sequence, is an integer parameter to be chosen that must be less than , and is defined as:

Where is the normalized hydrophobicity value for amino acid and is the normalized hydrophilicity value of amino acid .

### PSSM methods

21 feature extraction techniques were derived from the PSSM profiles of TE sequences. They were all implemented using the pssmpro [8] software package easily installable through pip (link: <https://pypi.org/project/pssmpro/>). Pssmpro is based on POSSUM [9], a tool to create various numerical descriptors of protein sequences.

PSSM profile of a sequence is constructed from multiple sequence alignment of the highest scoring hits in a BLAST search. PSI-BLAST program [10] was used to search NCBI’s UniRef50 database [11] with 5 iterations and cut-off E-value of 0.001 and create the PSSM profile for the TE sequences. The PSSM profile of an enzyme is an matrix where is the length of the enzyme sequence and 20 represents the number of amino acids that might be present at a specific position in a sequence. The -th element of the PSSM profile represents the score of the amino acid residue in the -th position of the enzyme sequence being mutated to amino acid type during the evolution process wherein a large value indicates a highly conserved position, and a small value indicates a weakly conserved position. The PSSM elements are usually scaled to a (0,1) range using the sigmoid function ,

where x is an element in the PSSM profile.

#### Amino Acid Composition using PSSM profiles (AAC-PSSM)

AAC-PSSM uses the numerical values of the PSSM matrix instead of the actual count of the amino acids as in traditional [AAC](#_Amino_Acid_Composition) to calculate composition of each amino acid. The PSSM profile of a protein sequence obtained using PSI-BLAST can be denoted as a matrix of length where is the length of the protein sequence. Thus, AAC-PSSM feature vector can be represented as:

Where is an element of the PSSM matrix corresponding to the -th position of the sequence and the -th amino acid type.

#### Di-Peptide Composition using PSSM profiles (DPC-PSSM)

DPC-PSSM uses the numerical values of the PSSM matrix instead of the actual count of di-peptides as in traditional DPC to calculate frequency of each dipeptide. The PSSM profile of a protein sequence obtained using PSI-BLAST can be denoted as a matrix of length where L is the length of the protein sequence. Thus, DPC-PSSM feature vector can be represented as:

Where is an element of the PSSM matrix corresponding to the -th position of the sequence and the -th amino acid type.

#### Amino Acid and Di-Peptide composition using PSSM profiles (AADP-PSSM)

AADP-PSSM is a concatenation of the feature vectors obtained using [AAC-PSSM](#_Amino_Acid_Composition_1) and [DPC-PSSM](#_Di-Peptide_Composition_using).

#### PSSM Auto-Covariance (PSSM-AC)

PSSM-AC applies auto-covariance transformation to each column of the PSSM matrix to measure the average correlation of the same property between residues separated by a distance of at most positions. For a protein , it can be denoted as:

Where is the length of the protein sequence, is an element of the PSSM matrix corresponding to the -th position of the sequence and the -th amino acid type, is the maximum distance between residues to consider and is pssm profile composition of amino acid throughout the length of the protein sequence as described in [AAC-PSSM](#_Amino_Acid_Composition_1).

#### PSSM Cross-Covariance (PSSM-CC)

PSSM-CC applies cross-covariance transformation to each column of the PSSM matrix to measure the average correlation of different properties between residues separated by a distance of at most D positions. For a protein , it can be denoted as:

Where is the length of the protein sequence, is an element of the PSSM matrix corresponding to the -th position of the sequence and the -th amino acid type, is the maximum distance between residues to consider and () is pssm profile composition of amino acid () throughout the length of the protein sequence as described in [AAC-PSSM](#_Amino_Acid_Composition_1). It should be noted that unlike [PSSM-AC](#_PSSM_Auto-Covariance_(PSSM-AC)), and are two different amino acid types.

#### Reduced PSSM (RPSSM)

RPSSM works with a reduced PSSM by grouping the 20 amino acids into 10 categories and compressing the PSSM into an matrix. At first, the elements in the PSSM matrix are mapped to the (0,1) range using sigmoid function. The grouping procedure is followed according to the rules of Li. et. al. [12] which is described as follows:

Where represents the new columns of the PSSM matrix and represents the columns in the PSSM matrix that correspond to amino acid . After creating a reduced PSSM profile, the pseudo composition of the amino acids and all the dipeptides in the protein sequence is calculated from the reduced profile as follows:

Where is the pseudo composition of amino acids, is an element of the reduced pssm matrix that corresponds to location in a sequence of length and grouped amino acid type , is the amino acid composition obtained from the reduced PSSM profile as defined in AAC and is the pseudo amino acid composition of all the dipeptides present in the protein sequence. The final feature vector is a concatenation of the two vectors and .

#### Tri-gram PSSM (tri-gram)

Tri-gram calculates the probabilities of the occurrence of a certain combination of amino acid triplets using the PSSM profile of the protein. At first, the elements in the PSSM matrix are mapped to the (0,1) range using sigmoid function. Then the feature vector is calculated as follows:

Where is an element of the pssm matrix that corresponds to location in a sequence of length and amino acid type .

#### EDP

EDP feature vectors are obtained from an Evolutionary Difference Matrix created by transforming the PSSM matrix into a dimensional ED-PSSM matrix where each element of the matrix represents a mutational difference between adjacent residues. To create the evolutionary difference matrix, the average of evolutionary score between adjacent residues are calculated as follows:

Where is the average evolutionary score between positions and , is the average evolutionary score between and and is an element of the PSSM matrix where denotes a position along the length of the protein sequence and denotes an amino acid. The evolutionary difference formula is subsequently calculated as follows:

The evolutionary difference formula is used to convert the PSSM matrix into ED-PSSM where an element of the ED-PSSM matrix is calculated as:

Where is an element of the ED-PSSM matrix and is the length of the protein sequence. The ED-PSSM matrix is used to calculate EDP feature vector as follows:

#### EEDP

The EEDP feature vector is also constructed from the ED-PSSM matrix described in [EDP](#_EDP). The feature vector is as follows:

Where is an element of the ED-PSSM matrix.

#### MEDP

MEDP feature vector is the concatenation of two feature vectors [EDP](#_EDP) and [EEDP](#_EEDP).

#### TPC-PSSM

TPC feature representation is obtained from the PSSM matrix as follows:

Where is an element of the PSSM matrix and is the length of the protein sequence.

#### AATP

AATP is the concatenation of two feature vectors [AAC-PSSM](#_Amino_Acid_Composition_1) and [TPC-PSSM](#_TPC-PSSM).

#### k-separated-bigrams

k-separated-bigrams model the relationship between non-adjacent amino acids separated by a distance of k-positions using the sequential evolution probabilities from the PSSM matrix of a protein sequence. It can be calculated as follows:

Where is an element of the pssm matrix and is the length of the protein sequence.

#### D-FPSSM

D-FPSSM is calculated from the FPSSM matrix obtained after filtering out all the negative scores from the PSSM matrix. After pre-processing, D-FPSSM is calculated as follows:

Where is an element of the FPSSM matrix.

#### S-FPSSM

S-FPSSM is also calculated from the FPSSM matrix described in [D-FPSSM](#_D-FPSSM) section as follows:

Where is an element in the FPSSM matrix that represents the -th position along the sequence length and -th amino acid and is an indicator function defined as:

#### PSE-PSSM

PSE-PSSM concatenates [AAC-PSSM](#_Amino_Acid_Composition_1) with a new set of descriptors adopted from pseudo-PSSM composition which can be calculated as:

Where is an element of the PSSM matrix that represents the score of the amino acid residue in -th position of the protein sequence being changed to amino acid type during the evolution process, is the length of the protein sequence and is the maximum distance between two residues whose PSSM scores are coupled.

#### DP-PSSM

DP-PSSM uses the PSSM profile to construct a feature vector as follows:

Where is an element of the PSSM matrix that represents the score of the amino acid residue in -th position of the protein sequence being changed to amino acid type during the evolution process, () is the number of positive (negative) terms in the column of the PSSM matrix, () is the number of positive (negative) terms obtained after calculating the difference where k denotes the spaces between amino acid residue positions, ranges from 1 to , being the length of the protein sequence under consideration and is the amino acid type to be considered.

#### PSSM-Composition

The PSSM-Composition feature vector is calculated as:

where is the amino acid residue at position of the sequence and is an element in the PSSM matrix.

#### Smoothed-PSSM

Smoothed-PSSM preprocesses the PSSM profile by incorporating information about the neighboring residues using a smoothing window size of w. Here, each row vector of a residue is represented as follows:

Where is an element of the smoothed pssm profile and is an element of the original pssm profile. After preprocessing, the feature vector of a protein is created by selecting residues based on a sliding window of size .

#### AB-PSSM

AB-PSSM constructs a feature set from PSSM matrix by dividing the matrix into 20 blocks where each block has number of rows equivalent to 5% of the length of protein sequence under consideration and 20 columns corresponding to the 20 types of amino acids. The feature vector is constructed as follows:

Where is the size of the block, is a vector of size 20 extracted from the PSSM profile corresponding to the -th row in the -th block.

#### RPM-PSSM

The RPM-PSSM vector is calculated as:

Where is the amino acid residue at position of the sequence and is an element in the PSSM matrix and is the number of amino acids of type in column of the PSSM matrix.

**Table S1**: 47 feature extraction techniques were used in EnZymClass. The feature extraction type, name, software used to create it, and literature from which it was adopted is provided in this table.

|  |  |  |  |
| --- | --- | --- | --- |
| Type | Name | Software Used | Literature Reference |
| Kernel | Spectrum Kernel | KeBABS | [2] |
| Kernel | Mismatch Kernel | KeBABS | [13] |
| Kernel | Gappy Pair Kernel | KeBABS | [14] |
| N-gram | Kmer | ngrampro | [4] |
| N-gram | GAA-kmer | ngrampro | NA |
| Physicochemical | AAC | ifeatpro | [15] |
| Physicochemical | CKSAAP | ifeatpro | [16] |
| Physicochemical | TPC | ifeatpro | [15] |
| Physicochemical | DPC | ifeatpro | [17] |
| Physicochemical | DDE | ifeatpro | [17] |
| Physicochemical | GAAC | ifeatpro | [18] |
| Physicochemical | CKSAAGP | ifeatpro | [6] |
| Physicochemical | GTPC | ifeatpro | [6] |
| Physicochemical | GDPC | ifeatpro | [6] |
| Physicochemical | Moran | ifeatpro | [19] |
| Physicochemical | Geary | ifeatpro | [20] |
| Physicochemical | NMBroto | ifeatpro | [21] |
| Physicochemical | CTDC | ifeatpro | [22]–[24] |
| Physicochemical | CTDT | ifeatpro | [22]–[24] |
| Physicochemical | CTDD | ifeatpro | [22]–[24] |
| Physicochemical | CTriad | ifeatpro | [25] |
| Physicochemical | KSCTriad | ifeatpro | [6] |
| Physicochemical | SOCNumber | ifeatpro | [6] |
| Physicochemical | QSOrder | ifeatpro | [6] |
| Physicochemical | PAAC | ifeatpro | [26] |
| Physicochemical | APAAC | ifeatpro | [26] |
| PSSM-based | AAC-PSSM | pssmpro | [27] |
| PSSM-based | DPC-PSSM | pssmpro | [27] |
| PSSM-based | AADP-PSSM | pssmpro | [27] |
| PSSM-based | PSSM-AC | pssmpro | [28] |
| PSSM-based | PSSM-CC | pssmpro | [29] |
| PSSM-based | RPSSM | pssmpro | [30] |
| PSSM-based | Tri-gram | pssmpro | [31] |
| PSSM-based | EDP | pssmpro | [32] |
| PSSM-based | EEDP | pssmpro | [32] |
| PSSM-based | MEDP | pssmpro | [32] |
| PSSM-based | TPC-PSSM | pssmpro | [33] |
| PSSM-based | AATP | pssmpro | [33] |
| PSSM-based | k-separated bigrams | pssmpro | [34] |
| PSSM-based | D-FPSSM | pssmpro | [35] |
| PSSM-based | S-FPSSM | pssmpro | [35] |
| PSSM-based | PSE-PSSM | pssmpro | [36] |
| PSSM-based | DP-PSSM | pssmpro | [37] |
| PSSM-based | PSSM-Composition | pssmpro | [38] |
| PSSM-based | Smoothed-PSSM | pssmpro | [39] |
| PSSM-based | AB-PSSM | pssmpro | [40] |
| PSSM-based | RPM-PSSM | pssmpro | [40] |

## Text S2

### Model training

EnZymClass can be trained using python’s numpy and scikit-learn modules [41], [42]. Model training and validation can be divided into five stages described below.

1. Random seed assignment and dataset division: EnZymClass requires a characterized dataset as input. It divides the characterized dataset (115 TE enzyme sequences labeled according to their corresponding substrate specificity category in this study) into training and validation set by a 75-25 percentage split. A random seed can be specified in EnZymClass to reproduce results. Changing the random seed will produce different training and validation sets, an event which will be used later to evaluate model performance.
2. Feature representation: The training and validation set of sequences needs to be encoded by different feature representation techniques before feeding them as input to EnZymClass. In the current work, we used feature representation methods described in the **Feature Extraction** section to transform TE sequences into 47 distinct feature vector representation. The distinct feature vectors of the training set of sequences were used to train 47 separate base learners operating on the same principle (PCA+SVM or PCA+NN or PCA+GBT).
3. Base model training**:** The base models accept the feature vector representation of protein sequences as input and predicts their functional attributes. They are trained using scikit-learn dedicated modules for Principal Component Analysis (PCA), Support Vector Classifier (SVC), Neural Network (NN) and Gradient Boosting Classifier (GBC). Each base model is a scikit-learn pipeline object consisting of PCA instance followed by the learning algorithm instance (SVC, NN, or GBC). The pipeline object was trained on the feature vector representation of the training set of TE sequences in this study. EnZymClass allows the hyperparameters of the base learners to be optimized using the GridSearchCV module of scikit-learn with a 5-fold cross-validation process.
4. Meta learner prediction: The trained base models in EnZymClass independently predict the functional attributes of protein sequences and pass on their predictions to the Meta Learner that uses a hard-voting based majority vote classifier to output the final prediction of the functional attribute. In the present study, the 47 base models trained only on the training set were used to independently predict the substrate specificity category of enzymes in both training and validation sets. The output predictions of these base learners were passed on to the meta learner. The parameter *k* of EnZymClass, representing the *k*-best base models to pass on to the meta learner, was chosen to be 5.
5. Model evaluation: EnZymClass allows model evaluation by providing methods to calculate four popular classification metrics, accuracy, precision, recall, and MCC. In the current study, the validation set accuracy, precision (on the medium-chain TEs) and recall (on the medium-chain TEs) of the 47 base learners and ensemble model were recorded. The entire Model Training procedure was repeated 10,000 times by varying the random seed specified initially. This resulted in different training and validation sets, thus affecting model performance and yielding a distribution of training and validation set accuracies, precision, and recall scores for 47 base learners and EnZymClass. The objective of evaluating our model multiple times by varying the training and validation set was to check its robustness to the training set. A parametric sweep of the ensemble model parameter *k* was performed to illustrate its effect on validation score.

### Model prediction

To predict uncharacterized protein sequences in a test set, the test sequences were converted to feature vectors using the same feature extraction techniques employed to encode sequences in the training set. The *N* base models delegated to each feature vector representation are subsequently trained and hyperparameter optimized using all the characterized protein sequences. The trained base models are used to independently predict the functional attributes of test set proteins. The output predictions of the *k* best-performing base learners are passed on to a meta learner that uses a hard-voting based majority vote classifier to output the final prediction of the functional attribute. In the present work, we followed the above procedure to obtain substrate specificity predictions of uncharacterized TE sequences. The parameter *k* was selected based on the results of the parametric sweep study discussed in the **Model Training** section.

Diagram

Description automatically generated

**Figure S1**: EnZymClass pipeline is described here. Unique feature representation technique designated Base Learners get independently trained and predict functional attributes of protein sequences. The independent predictions are collected by the Meta Learner that outputs the final model prediction by calculating the majority vote of the predictions made by each individual Base Learner. EnZymClass requires a labelled characterized set of sequences as input. Users can also provide an unlabeled test set and optimized hyperparameters for each Base Learner as optional arguments. Training and validation sets are created from subsets of the characterized “training” set. EnZymClass outputs model predictions for the training set, validation set and test set (if provided).

### Computational study workflow

In the current study, we employed EnZymClass to predict substrate specificity of plant acyl-ACP TEs by training it on a small, unbalanced TE dataset with high sequence similarity between the enzyme sequences. EnZymClass is specifically built to tackle the challenges of high-dimensionality, small-sized and imbalanced datasets, correlated features, and high sequence similarity among proteins at every stage of its pipeline. We used 47 alignment-free feature extraction techniques, proven to be effective in multiple application areas of protein sequence classification, to numerically encode TE sequences such that EnZymClass can effectively extract as much information from primary sequences as realistically possible. To prevent overfitting, the feature vectors generated through the extraction process were decomposed into lower dimensional and linearly uncorrelated features using Principal Component Analysis. The lower dimensional and decomposed set of feature vectors were used to train individual base learners and independently predict substrate specificity of TEs. An ensemble method was used to circumvent the dataset imbalance problem and retain high prediction accuracy. The generalizability of EnZymClass was assessed through a rigorous model validation strategy where we simulated 10,000 different versions of training and validation datasets from the characterized set of TE sequences. We recorded the performance of the framework on these validation datasets using three popular classification metrics, accuracy, precision, and recall.

EnZymClass’ novel pipeline and model architecture allows it to overcome challenges posed by small, characterized protein sequence datasets for the purpose of attaining high prediction accuracies. It employs 47 alignment-free descriptors of protein sequences, the maximum number of encoders used by any protein classification algorithm known, with the ability to automatically select the encoders best suited for a specific protein classification task. Intuitively, the ability of EnZymClass to select the best set of feature descriptors should allow it to generalize across protein classification datasets covering a wide range of application areas in computational biology. It is worth noting that although deep learning algorithms such as CNNs or RNNs can automatically build features descriptive of a specific classification task, they require characterized datasets ranging from thousands to millions of instances. unlike EnZymClass which can work with as few as a hundred training instances as illustrated in this study. Moreover, unlike similar ensemble learning algorithms described in [43]–[48], EnZymClass includes an in-built decomposition technique to tackle issues related to high-dimensionality and is capable of self-tuning model hyperparameters, a phenomenon that often leads to better and more robust model performance. Additionally, it provides the user with the flexibility of choosing the base learner training algorithm among SVM, GBC and NN.

A picture containing text, parking, sign

Description automatically generated

**Figure S2**: a) outlines the entire workflow of TE substrate specificity prediction method. Both characterized and uncharacterized TE sequences are encoded into 47 unique training and test feature sets. In the Model Training phase, the training feature sets are used to optimize hyperparameters of EnZymClass in addition to evaluating model predictions on a validation set created by splitting the original training set into two segments. A parametric sweep of the EnZymClass hyperparameter k, the number of best base learners that passes on their predictions to the meta learner, is also performed. Finally, the optimized hyperparameters along with the entire training set is used to train EnZymClass and predict substrate specificity of uncharacterized TE sequences in the Model Prediction phase; b) describes the Base Learner Hyperparameter Optimization component of the Model Training phase. The unique set of features are divided into different training and validation sets, 1000 times. Each time, EnZymClass is trained, and the base learners are hyperparameter optimized using a training set. The optimized hyperparameters are stored for each run. At the end of 1000 runs, the most frequent hyperparameters among the list of hyperparameters for each base learner are selected as the chosen hyperparameters; c) illustrates the Model Evaluation component of the Model Training phase. The unique set of features are divided into different training and validation sets, 10000 times. Each time, EnZymClass is trained using the training set and the optimized hyperparameters obtained in the Base Learner Hyperparameter Optimization stage. The resulting model predictions of the Base and Meta Learners in EnZymClass are noted and they are used to calculate classification metrics that indicate model performance. The list of metrics for each run is stored and a distribution of classification metrics are returned after 10,000 runs; d) depicts the Model Prediction phase. The entire Training feature set along with the previously optimized hyperparameters are used to train EnZymClass which predicts the substrate specificity of the test set enzymes.

**Table S 2**: The hyperparameters of each base learner are dependent on the learning algorithm used to train it. The learning algorithm-dependent hyperparameters are displayed here.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Learning Algorithm | Hyperparameter I | Hyperparameter II | Hyperparameter III | Hyperparameter IV |
| SVM | PCA Components | Kernel | Regularization C | Gamma |
| NN | PCA Components | Hidden layer size | Learning rate | Regularization |
| GBC | PCA Components | Number of estimators | Learning rate | Tree max depth |

# Text S3

## Mathematical formulation of performance metrics

Accuracy score for a multi-class classification problem is defined as:

Where is the predicted value for sample *i*, is the corresponding true sample, *N* is the number of samples, and **1** is an indicator function that is equals to 1 if a certain condition is true and 0 otherwise.

Precision score for a class is defined as:

Where is the number of true positives for the *i*-th class and is the number of false positives for the *i*-th class.

Recall score for a class *i* is defined as:

Where is the number of false negatives recorded for the *i*-th class.

Matthew’s Correlation Coefficient for a multiclass classification problem is defined as:

Where, or the total number of instances correctly predicted across all *k* classes, or the total number of instances, or the number of times class *k* actually occurred, or the number of times class *k* was predicted, and *C* is a confusion matrix of size *k* x *k*.

## Feature extraction techniques comparison

We compared the performance of 47 different base models, each trained on a unique feature encoded representation of TE sequences, on validation set using three popular classification performance metrics to detect the feature extraction techniques which can most successfully discriminate between enzyme substrate specificity classes. The three metrics were 1) accuracy score on all three categories of TE substrate specificity, 2) precision score on medium-chain TEs and 3) recall score on medium-chain TEs. Our validation scheme produced a distribution of model metrics through 10,000 simulations of our model evaluation experiment by varying the training and validation dataset during each simulation as discussed in the **Model Training** section. A list of the 47 feature encoding techniques ranked by their mean accuracy scores on TE substrate specificity prediction is given in **Table S3**.

**Table S3**: A list of three performance metrics for the 47 base models ranked by their mean accuracy score on varying validation datasets generated through 10000 simulations of model evaluation experiment is shown. The table presents the mean precision and recall scores obtained on the medium-chain TEs of varying validation datasets apart from the mean accuracy score achieved on all three substrate specificity categories of TEs by the 47 base models along with their feature extraction technique.

|  |  |  |  |
| --- | --- | --- | --- |
| Model Feature Extraction | Mean Accuracy | Mean Precision | Mean Recall |
| **spectrumKernel** | 0.87 | 0.85 | 0.77 |
| **gappyKernel** | 0.87 | 0.86 | 0.77 |
| **cksaap** | 0.87 | 0.86 | 0.77 |
| **ksctriad** | 0.86 | 0.85 | 0.76 |
| **moran** | 0.87 | 0.85 | 0.76 |
| **dpc** | 0.87 | 0.85 | 0.76 |
| **ctriad** | 0.86 | 0.85 | 0.76 |
| **geary** | 0.87 | 0.84 | 0.75 |
| **cksaagp** | 0.82 | 0.84 | 0.75 |
| **nmbroto** | 0.87 | 0.82 | 0.75 |
| **gtpc** | 0.84 | 0.84 | 0.75 |
| **dde** | 0.85 | 0.85 | 0.75 |
| **tpc** | 0.81 | 0.84 | 0.74 |
| **pssm\_composition** | 0.84 | 0.86 | 0.73 |
| **kmer** | 0.79 | 0.82 | 0.73 |
| **rpm\_pssm** | 0.83 | 0.84 | 0.73 |
| **s\_fpssm** | 0.84 | 0.84 | 0.73 |
| **gaakmer** | 0.80 | 0.82 | 0.71 |
| **qsorder** | 0.81 | 0.82 | 0.70 |
| **ctdc** | 0.77 | 0.80 | 0.69 |
| **mismatchKernel** | 0.72 | 0.86 | 0.69 |
| **socnumber** | 0.78 | 0.77 | 0.69 |
| **paac** | 0.80 | 0.82 | 0.68 |
| **apaac** | 0.80 | 0.82 | 0.68 |
| **aac** | 0.80 | 0.81 | 0.67 |
| **ctdt** | 0.77 | 0.78 | 0.67 |
| **rpssm** | 0.65 | 0.84 | 0.66 |
| **medp** | 0.64 | 0.84 | 0.65 |
| **eedp** | 0.64 | 0.84 | 0.65 |
| **aadp\_pssm** | 0.62 | 0.89 | 0.65 |
| **dpc\_pssm** | 0.62 | 0.89 | 0.65 |
| **d\_fpssm** | 0.63 | 0.83 | 0.65 |
| **gdpc** | 0.59 | 0.92 | 0.64 |
| **tpc\_pssm** | 0.72 | 0.77 | 0.64 |
| **ctdd** | 0.73 | 0.73 | 0.64 |
| **dp\_pssm** | 0.61 | 0.85 | 0.63 |
| **pssm\_ac** | 0.70 | 0.76 | 0.63 |
| **pssm\_cc** | 0.69 | 0.78 | 0.63 |
| **aatp** | 0.60 | 0.85 | 0.62 |
| **k\_separated\_bigrams\_pssm** | 0.60 | 0.83 | 0.62 |
| **pse\_pssm** | 0.61 | 0.85 | 0.62 |
| **aac\_pssm** | 0.59 | 0.84 | 0.61 |
| **smoothed\_pssm** | 0.59 | 0.82 | 0.61 |
| **edp** | 0.59 | 0.85 | 0.61 |
| **ab\_pssm** | 0.58 | 0.80 | 0.60 |
| **tri\_gram\_pssm** | 0.66 | 0.65 | 0.59 |
| **gaac** | 0.54 | 0.94 | 0.58 |

## Base learner assessment

The designed ensemble framework allows the base learners to be trained on any one of the three learning algorithms, SVM, NN and GBC. We assessed the effect of training all base models in the ensemble using each of the three learning algorithms on the ensemble model’s performance. Our base learner assessment strategy followed the same evaluation technique as discussed **Model Training**, which created a distribution of 10,000 model performance metrics for every base learning algorithm. The three metrics used to judge the ensemble model performance were mean accuracy score, mean precision score on the medium-chain TEs and mean recall scores on the medium-chain TEs. The ensemble model with SVM as the base learner training algorithm outperformed both NN and GBC on mean accuracy and mean precision (on medium-chain TEs) but was beaten by the other learning algorithms in terms of mean recall score (on medium-chain TEs). The performance scores of the ensemble model trained separately on each of the three learning algorithms is given in **Table S4**.

**Table S4**: The ensemble framework is capable of using three different learning algorithms to train its base models. The performance of the ensemble depends on the learning algorithm used. The mean accuracy, mean precision score (on medium-chain TEs) and mean recall score (on medium-chain TEs) achieved by the ensemble when it was trained using each of the three learning algorithms, SVM, NN and GBC is listed here.

|  |  |  |  |
| --- | --- | --- | --- |
| Model Learning Algorithm | Mean Accuracy | Mean Precision | Mean Recall |
| SVM | 0.80 | 0.88 | 0.89 |
| GBC | 0.79 | 0.82 | 0.92 |
| NN | 0.79 | 0.81 | 0.94 |

## Experimental Materials and Supplemental Data

**Table S5**: A list of *E*. coli strains and expression plasmids used in this study.

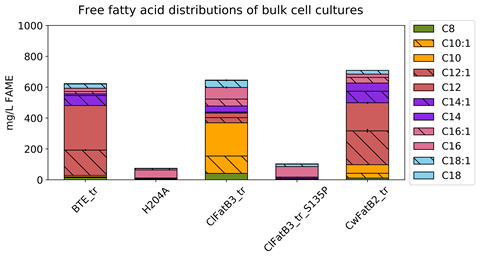
|  |  |  |
| --- | --- | --- |
| **Strain/plasmid** | **Genotype** | **Source** |
| E. coli K12 MG1655 CGSG | F - λ - ilvGrfb-50 rph-1 | CGSG |
| RL08ara | K-12 MG1655 Δ*araBAD* Δ*fadD* | [49] |
| pTRC99a-BbvCI | ptrc promoter, pBR322 origin, AmpR, BbvCI nickase cut site incorporated to facilitate mutagenesis described in Wrenbeck *et al.* [50] | This study |
| pTRC99a-BbvCI-BTE | pTRC99a vector with *Umbellularia californica* UcFatB1 gene truncated at TLEWK | This study |
| pTRC99a-BbvCI-BTE\_trunc | pTRC99a vector with *Umbellularia californica* UcFatB1 gene truncated at WKPKP to align with CupTE-M4 truncation (ΔA54) described in Hernández Lozada *et al.* [51] | This study |
| pTRC99a-BTE\_H204A | pTRC99a vector with *Umbellularia californica* UcFatB1 gene with catalytic histidine swapped for alanine. Used as negative control | [49] |
| pTRC99a-BbvCI-ClFatB3 | pTRC99a vector with *Cuphea lanceolata* ClFatB3 gene truncated at LPDWS | This study |
| pTRC99a-BbvCI-ClFatB3\_trunc | pTRC99a vector with *Cuphea lanceolata* ClFatB3 gene truncated at LDRKS to align with CupTE-M4 truncation described in Hernández Lozada *et al.* [51] | This study |
| pTRC99a-BbvCI-ClFatB3\_trunc\_M1 | pTRC99a vector with *Cuphea lanceolata* ClFatB3 gene truncated at LDRKS to align with CupTE-M4 truncation and D10S point mutation described in Hernández Lozada *et al.* [51]. | This study |
| pTRC99a-BbvCI-ClFatB3\_trunc\_M2 | pTRC99a vector with *Cuphea lanceolata* ClFatB3 gene truncated at LDRKS to align with CupTE-M4 truncation and I47M point mutation described in Hernández Lozada *et al.* [51] | This study |
| pTRC99a-BbvCI-ClFatB3\_trunc\_M3 | pTRC99a vector with *Cuphea lanceolata* ClFatB3 gene truncated at LDRKS to align with CupTE-M4 truncation and both D10S and I47M point mutations described in Hernández Lozada *et al.* [51] | This study |
| pTRC99a-BbvCI-ClFatB3-2 | pTRC99a vector with *Cuphea lanceolata* alternative ClFatB3 gene truncated at LPDWS and with the S135P substitution in binding pocket | This study |
| pTRC99a-BbvCI-ClFatB3-2\_trunc | pTRC99a vector with *Cuphea lanceolata* alternative ClFatB3 gene truncated at LDRKS to align with CupTE-M4 truncation described in Hernández Lozada *et al.* [51]and with proline substitution in binding pocket | This study |
| pTRC99a-BbvCI-ClFatB4 | pTRC99a vector with *Cuphea lanceolata* ClFatB4 gene truncated at LPDWS | This study |
| pTRC99a-BbvCI-CpaFatB1 | pTRC99a vector with *Cuphea paucipetala* CpaFatB1 gene truncated at LPDWS | This study |
| pTRC99a-BbvCI-CpaFatB2A | pTRC99a vector with *Cuphea paucipetala* CpaFatB2A gene truncated at LPDWS | This study |
| pTRC99a-BbvCI-CpaFatB3 | pTRC99a vector with *Cuphea paucipetala* CpaFatB3 gene truncated at LPDWS | This study |
| pTRC99a-BbvCI-CwFatB2 | pTRC99a vector with *Cuphea wrightii* CwFatB2 gene truncated at LPDWS | This study |
| pTRC99a-BbvCI-CwFatB2\_trunc | pTRC99a vector with *Cuphea wrightii* CwFatB2 gene truncated at LDRKS to align with CupTE-M4 truncation described in Hernández Lozada *et al.* [51] | This study |

**Table S6**: Fatty acid methyl ester distributions and titers obtained from derivatization of bulk stationary phase culture following *in vivo* culture experiments. The GenBank Accession ID is listed for the mRNA translation of each TE gene tested. The percentage of each chain length in the overall distribution and the overall titer are reported as averages among three biological replicates, plus or minus the standard error of the mean. All data points were collected in this study.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Construct** | **Thioesterase source organism** | **GenBank ID** | **% C8** | **% C10** | **% C10:1** | **% C12** | **% C12:1** | **% C14** | **% C14:1** | **% C16** | **% C16:1** | **% C18:0** | **% C18:1** | **mg/L** |
| pTRC99a-BbvCI-BTE | *Umbellularia californica* | AAA34215.1 | 0.59±0.02 | 0.29±0.01 | 1.13±0.02 | 14.24±0.45 | 66.4±0.28 | 7.83±0.08 | 2.71±0.05 | 1.53±0.15 | 1.56±0.07 | 3.59±0.18 | 0.13±0.0 | 900.8±22.9 |
| pTRC99a-BbvCI-BTE\_trunc | *Umbellularia californica* | AAA34215.1 | 2.39±0.11 | 0.67±0.02 | 1.53±0.03 | 26.33±0.21 | 46.27±0.38 | 10.4±0.23 | 1.78±0.09 | 2.42±0.08 | 3.33±0.25 | 4.5±0.09 | 0.36±0.03 | 623.4±9.30 |
| pTRC99a-BTE\_H204A | *Umbellularia californica* | AAA34215.1 | 5.82±0.45 | 0.0±0.0 | 0.0±0.0 | 0.0±0.0 | 3.45±0.28 | 0.19±0.19 | 5.72±0.21 | 6.98±1.04 | 57.27±2.8 | 16.99±2.23 | 3.58±0.14 | 89.16±9.75 |
| pTRC99a-BbvCI-ClFatB3 | *Cuphea lanceolata* | ADR51157.1 | 1.66±0.03 | 3.1±0.05 | 17.72±0.14 | 2.17±0.05 | 4.11±0.02 | 0.58±0.01 | 10.58±0.05 | 6.72±0.04 | 39.06±0.14 | 13.19±0.14 | 1.1±0.01 | 355.0±27.3 |
| pTRC99a-BbvCI-ClFatB3\_trunc | *Cuphea lanceolata* | ADR51157.1 | 6.42±0.14 | 17.45±0.36 | 33.28±0.66 | 5.29±0.18 | 4.48±0.04 | 1.02±0.02 | 6.07±0.09 | 6.92±0.16 | 11.67±0.45 | 7.11±0.5 | 0.3±0.01 | 646.6±13.2 |
| pTRC99a-BbvCI-ClFatB3\_trunc\_M1 | *Cuphea lanceolata* | ADR51157.1 | 4.52±0.05 | 18.44±0.36 | 34.56±0.45 | 6.08±0.18 | 4.4±0.03 | 1.1±0.03 | 5.75±0.05 | 7.75±0.08 | 9.52±0.29 | 7.76±0.59 | 0.13±0.0 | 736.3±8.18 |
| pTRC99a-BbvCI-ClFatB3\_trunc\_M2 | *Cuphea lanceolata* | ADR51157.1 | 3.17±0.01 | 12.2±0.16 | 32.09±0.16 | 4.37±0.06 | 4.78±0.0 | 0.85±0.01 | 7.19±0.05 | 6.63±0.1 | 17.45±0.21 | 11.01±0.26 | 0.25±0.01 | 581.0±5.41 |
| flpTRC99a-BbvCI-ClFatB3\_trunc\_M3 | *Cuphea lanceolata* | ADR51157.1 | 4.97±0.26 | 20.9±0.66 | 33.75±0.58 | 7.12±0.31 | 4.15±0.12 | 1.24±0.06 | 5.19±0.12 | 6.88±0.03 | 8.82±0.31 | 6.62±0.43 | 0.37±0.04 | 532.8±30.3 |
| pTRC99a-BbvCI-ClFatB3-2 | *Cuphea lanceolata* | CAB60830.1 | 1.4±0.04 | 0.0±0.0 | 0.52±0.02 | 0.0±0.0 | 1.43±0.18 | 0.59±0.04 | 6.7±0.68 | 9.2±0.31 | 62.97±1.17 | 15.8±0.72 | 1.39±0.13 | 122.6±2.10 |
| pTRC99a-BbvCI-ClFatB3-2\_trunc | *Cuphea lanceolata* | CAB60830.1 | 1.62±0.08 | 0.0±0.0 | 0.76±0.04 | 0.0±0.0 | 1.87±0.06 | 0.18±0.18 | 6.27±0.06 | 6.71±1.23 | 65.56±1.94 | 15.32±0.9 | 1.72±0.16 | 103.5±4.49 |
| pTRC99a-BbvCI-ClFatB4 | *Cuphea lanceolata* | ADR51158.1 | 0.66±0.08 | 0.0±0.0 | 0.07±0.03 | 0.03±0.03 | 1.52±0.03 | 1.29±0.02 | 49.05±0.11 | 29.64±0.27 | 7.87±0.16 | 9.77±0.11 | 0.11±0.05 | 758.3±53.8 |
| pTRC99a-BbvCI-CpaFatB1 | *Cuphea paucipetala* | ADR51161.1 | 1.01±0.12 | 0.0±0.0 | 0.18±0.01 | 0.0±0.0 | 0.61±0.03 | 0.2±0.01 | 11.29±0.22 | 15.45±0.37 | 48.98±0.33 | 21.48±0.23 | 0.8±0.02 | 363.0±22.0 |
| pTRC99a-BbvCI- CpaFatB2A | *Cuphea paucipetala* | ADR51162.1 | 0.6±0.06 | 0.0±0.0 | 0.11±0.0 | 0.0±0.0 | 0.44±0.04 | 0.14±0.0 | 17.87±0.58 | 23.72±1.11 | 34.5±1.1 | 22.12±0.56 | 0.49±0.02 | 631.1±14.3 |
| pTRC99a-BbvCI-CpaFatB3 | *Cuphea paucipetala* | ADR51163.1 | 0.13±0.0 | 0.0±0.0 | 0.08±0.01 | 0.0±0.0 | 0.69±0.02 | 0.46±0.03 | 40.04±0.7 | 25.13±0.71 | 16.37±0.8 | 16.76±0.63 | 0.33±0.01 | 904.0±43.4 |
| pTRC99a-BbvCI-CwFatB2 | *Cuphea wrightii* | AAC49784.1 | 1.61±0.03 | 3.36±0.02 | 6.99±0.1 | 25.4±0.18 | 26.06±0.12 | 11.91±0.06 | 10.84±0.11 | 5.95±0.05 | 4.06±0.01 | 3.81±0.01 | 0.0±0.0 | 704.7±8.44 |
| pTRC99a-BbvCI-CwFatB2\_trunc | *Cuphea wrightii* | AAC49784.1 | 1.59±0.11 | 4.43±0.11 | 7.76±0.27 | 30.93±0.49 | 25.65±0.09 | 10.57±0.21 | 7.49±0.17 | 5.14±0.2 | 2.98±0.14 | 3.46±0.17 | 0.0±0.0 | 709.4±20.4 |

**Table S7**:Fatty acid methyl ester distributions and titers obtained from derivatization of culture supernatants following *in vivo* culture experiments. The percentage of each chain length in the overall distribution and the overall titer are reported as averages among three biological replicates, plus or minus the standard error of the mean. All data points were collected in this study.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Construct** | **Thioesterase source organism** | **GenBank ID** | **% C8** | **% C10** | **% C10:1** | **% C12** | **% C12:1** | **% C14** | **% C14:1** | **% C16** | **% C16:1** | **% C18:0** | **% C18:1** | **mg/L** |
| pTRC99a-BbvCI-BTE | *Umbellularia californica* | AAA34215.1 | 0.6±0.02 | 0.4±0.0 | 1.5±0.02 | 19.42±0.37 | 66.57±0.43 | 7.63±0.14 | 1.4±0.04 | 0.65±0.06 | 0.45±0.03 | 1.29±0.07 | 0.08±0.04 | 661.3±26.9 |
| pTRC99a-BTE\_H204A | *Umbellularia californica* | AAA34215.1 | 38.83±3.91 | 0.0±0.0 | 6.28±0.58 | 0.0±0.0 | 21.35±2.76 | 0.0±0.0 | 7.95±0.37 | 1.83±1.83 | 18.1±0.31 | 2.52±2.52 | 3.15±3.15 | 8.61±0.84 |
| pTRC99a-BbvCI-ClFatB3 | *Cuphea lanceolata* | ADR51157.1 | 4.77±0.01 | 8.55±0.06 | 48.38±0.06 | 5.78±0.09 | 9.22±0.08 | 1.11±0.02 | 4.92±0.06 | 4.15±0.05 | 8.24±0.02 | 4.29±0.09 | 0.59±0.3 | 116.4±1.80 |
| pTRC99a-BbvCI-ClFatB3-2 | *Cuphea lanceolata* | CAB60830.1 | 16.04±0.57 | 0.0±0.0 | 6.26±0.22 | 0.0±0.0 | 6.78±0.18 | 0.0±0.0 | 11.51±1.17 | 11.37±0.56 | 35.56±1.89 | 12.49±0.21 | 0.0±0.0 | 9.73±0.13 |
| pTRC99a-BbvCI-CwFatB2 | *Cuphea wrightii* | AAC49784.1 | 2.58±0.09 | 5.55±0.09 | 10.85±0.24 | 37.2±0.16 | 23.52±0.06 | 9.89±0.17 | 4.34±0.04 | 3.05±0.06 | 1.43±0.05 | 1.59±0.05 | 0.0±0.0 | 444.2±21.3 |



**Figure S3**: The effect of TE homolog on free fatty acid distribution in bulk cultures of RL08*ara* *E. coli* cells. Each TE was truncated at the location where the N-terminus aligned with the ΔA54 truncation in CupTE. The distribution from cells expressing the California bay laurel TE (BTE) and a catalytically inactive BTE variant (H204A) are shown as a positive and negative control, respectively. The effect of the truncation was most beneficial to the ClFatB3 enzyme, resulting in a 3.3-fold increase in combined decanoic and decenoic acid titers. The CwFatB2\_tr variant displayed a 5% increase of dodecanoic acid in the free fatty acid distribution when compared to the non-truncated sequence but did not exhibit an improvement in overall titer. The ClFatB3-2 TE sequence was also truncated to determine if its inactivity could be overcome by the N-terminal modification. However, no change was observed. Bar height represents the average titer obtained from biological triplicates, and error bars represent the standard error of the mean.

CupTE -MVAAAASSACFPVPSPGASPKPGKLGNWSSSLSPSLKPKSIPNGGFQVKANASAHPKAN 59

BTE -MATTSLASAFCSMKA-----------------------VMLARDGRG--------MKPR 28

ClFatB3 -MVAAAATSAFFPVPAPGTSPKPGKSGNWPSSLSPTFKPKSIPNAGFQVKANASAHPKAN 59

ClFatB3-2 -MVAAAATSAFFPVPAPGTSPKPGKSGNWPSSLSPTFKPKSIPNAGFQVKANASAHPKAN 59

CwFatB2 MVVAAAASSAFFPVPAPRPTPKPGKFGNWPSSLSQPFKPKSNPNGRFQVKANVSPHPKAN 60

CupTE GSAVTLKSGSLNTQEDTL-SSSPPPRAFFNQLPDWSMLLTAITTVFVAPEKRWTMFDRK- 117

BTE SSDLQLRAGNAPTSLKMINGTKFSYTESLKRLPDWSMLFAVITTIFSAAEKQWTNLEWKP 88

ClFatB3 GSAVNLKSGSLNTQEDT--SSSPPPRAFLNQLPDWSMLLTAITTVFVAAEKQWTMLDRK- 116

ClFatB3-2 GSAVNLKSGSLNTQEDT--SSSPPPRAFLNQLPDWSMLLTAITTVFVAAEKQWTMLDRK- 116

CwFatB2 GSAVSLKSGSLNTLEDP--PSSPPPRTFLNQLPDWSRLRTAITTVFVAAEKQFTRLDRK- 117

1 2

CupTE -SKRPNMLMDSFGLERVVQDGLVFRQSFSIRSYEICADRTASIETVMNHVQETSLNQCKS 176

BTE KPKLPQLLDDHFGL-----HGLVFRRTFAIRSYEVGPDRSTSILAVMNHMQEATLNHAKS 143

ClFatB3 -SKRPDMLVDSVGLKSIVRDGLVSRQSFLIRSYEIGADRTASIETLMNHLQETSINHCKS 175

ClFatB3-2 -SKRPDMLVDSVGLKSIVRDGLVSRQSFLIRSYEIGADRTASIETLMNHLQETSINHCKS 175

CwFatB2 -SKRPDMLVDWFGSETIVQDGLVFRERFSIRSYEIGADRTASIETLMNHLQDTSLNHCKS 176

3 4

CupTE IGLLDDGFGRSPEMCKRDLIWVVTRMKIMVNRYPTWGDTIEVSTWLSQSGKIGMGRDWLI 236

BTE VGILGDGFGTTLEMSKRDLMWVVRRTHVAVERYPTWGDTVEVECWIGASGNNGMRRDFLV 203

ClFatB3 LGLLNDGFGRTPGMCKNDLIWVLTKMQIMVNRYPTWGDTVEINTWFPQSGKIGMASDWLI 235

ClFatB3-2 LGLLNDGFGRTPGMCKNDLIWVLTKMQIMVNRYPTWGDTVEINTWFSQSGKIGMASDWLI 235

CwFatB2 VGLLNDGFGRTPEMCTRDLIWVLTKMQIVVNRYPTWGDTVEINSWFSQSGKIGMGREWLI 236

5

CupTE SDCNTGEILVRATSVYAMMNQKTRRFSKLPHEVRQEFAPHFLDSPPAIEDNDGKLQKFDV 296

BTE RDCKTGEILTRCTSLSVLMNTRTRRLSTIPDEVRGEIGPAFIDNVAVKDDEIKKLQKLND 263

ClFatB3 SDCNTGEILIRATSVWAMMNQKTRRFSRLPYEVRQELTPHFVDSPHVIEDNDQKLHKFDV 295

ClFatB3-2 SDCNTGEILIRATSVWAMMNQKTRRFSRLPYEVRQELTPHFVDSPHVIEDNDQKLHKFDV 295

CwFatB2 SDCNTGEILVRATSAWAMMNQKTRRFSKLPCEVRQEIAPHFVDAPPVIEDNDRKLHKFDV 296

CupTE KTGDSIRKGLTPGWYDLDVNQHVSNVKYIGWILESMPTEVLETQELCSLTLEYRRECGRD 356

BTE STADYIQGGLTPRWNDLDVNQHVNNLKYVAWVFETVPDSIFESHHISSFTLEYRRECTRD 323

ClFatB3 KTGDSIRKGLTPRWNDLDVNQHVSNVKYIGWILESMPIEVLETQELCSLTVEYRRECGMD 355

ClFatB3-2 KTGDSIRKGLTPRWNDLDVNQHVSNVKYIGWILESMPIEVLETQELCSLTVEYRRECGMD 355

CwFatB2 KTGDSICKGLTPGWNDFDVNQHVSNVKYIGWILESMPTEVLETQELCSLTLEYRRECGRE 356

CupTE SVLESVTSMDPSKVGDRFQYRHLLRLEDGADIMKGRTEWRPKNAGTNGAISTGKT----- 411

BTE SVLRSLTTVSGGSSEAGLVCDHLLQLEGGSEVLRARTEWRPKLTDSFRGISVIPAEPRV- 382

ClFatB3 SVLESVTAVDPSENGGRSQYKHLLRLEDGTDIVKSRTEWRPKNAGTNGAISTSTAKTSNG 415

ClFatB3-2 SVLESVTAVDPSENGGRSQYKHLLRLEDGTDIVKSRTEWRPKNAGTNGAISTSTAKTSNG 415

CwFatB2 SVVESVTSMNPSKVGDRSQYQHLLRLEDGADIMKGRTEWRPKNAGTNRAIST-------- 408

CupTE ---- 411

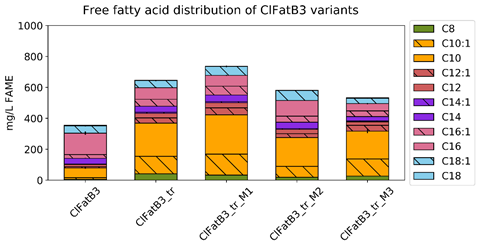
BTE ---- 382

ClFatB3 NSAS 419

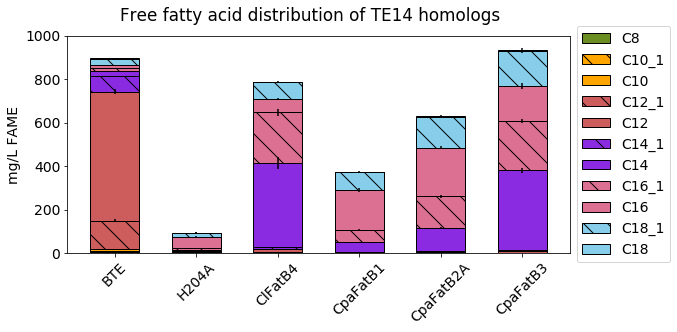
ClFatB3-2 NSAS 419

CwFatB2 ---- 408

**Figure S4**: Multiple sequence alignment of BTE, ClFatB3, ClFatB3-2, and CwFatB2 to CpFatB1 generated by Clustal Omega. The standard signal peptide truncation used for expressing TE in *E. coli* is indicated at (1) and the location of the CupTE ΔA54 truncation is indicated at (2). The location of the CupTE N28S and I65M point mutations which were incorporated into the ClFatB3 gene are denoted at (3) and (4), respectively. The S135P residue variation between ClFatB3 and ClFatB3-2 is marked at (5).



**Figure S5**:The effect of ClFatB3 variant on free fatty acid distribution in bulk cultures of RL08*ara* *E. coli* cells. Each TE was truncated at the location where the N-terminus aligned with the ΔA54 truncation in CupTE. Bar height represents the average titer obtained from biological triplicates, and error bars represent the standard error of the mean.



**Figure S6**: The effect of various TE homologs from the *Cuphea* genus on free fatty acid distribution in bulk cultures of RL08*ara* *E. coli*cells. The distribution from cells expressing the California bay laurel TE (BTE) and a catalytically inactive BTE variant (H204A) are shown as a positive and negative control, respectively. Bar height represents the average titer obtained from biological triplicates, and error bars represent the standard error of the mean.

\*Some studies report butyric and hexanoic acid. These species are included in the C8 and below column.

\*\*TE for which approximate but not absolute distributions are known.

N.R. denotes not reported.

**Table S8**: Fatty acid methyl ester distributions and titers obtained from various *in vivo* culture experiments used to train EnZymClass. The percentage of each chain length in the overall distribution and the overall titer are listed.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Thioesterase source organism and identifier** | **GenBank ID** | **% C8** | **% C10** | **% C10:1** | **% C12** | **% C12:1** | **% C14** | **% C14:1** | **% C16** | **% C16:1** | **% C18:0** | **% C18:1** | **mg/L** | **Source** |
| *A. hypogaea l.* (AhFatA) | ADB79567 | 0 | 0 | 0 | 4 | 0 | 8 | 0 | 45 | 14 | 2 | 6 | 6 | [52] |
| *Arabidopsis thaliana* | AEE28300.1 | 0 | 0 | N.R. | 0 | 0 | 33 | N.R. | 40 | 0 | 25 | 0 | 711 | [53] |
| *Auxenochlorella protothecoides* | KFM28838.1 | 1 | 0 | N.R. | 0 | 0 | 18 | N.R. | 73 | 0 | 6 | 1 | 399 | This study |
| *Brassica juncea* (BjFatB1) | ACR56792.1 | 0 | 0 | 0 | 0 | 0 | 4 | 0 | 36 | 3 | 53 | 1 | N.R | [54] |
| *Brassica juncea* (BjFatB2) | ACR56793.1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 59 | 3 | 32 | 4 | N.R | [54] |
| *Brassica juncea* (BjFatB3) | ACR56794.1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 51 | 2 | 43 | 2 | N.R | [54] |
| *Brassica juncea* (BjFatB4) | ACR56795.1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 62 | 3 | 27 | 4 | N.R | [54] |
| *Cinnamomum camphorum\*\** | AAC49151.1 | 0 | 0 | 0 | 20 | 0 | 80 | 0 | 0 | 0 | 0 | 0 | 88 | [55] |
| *Cocos nucifera* (CnFatB1) | AEM72522.1 | 14 | 1 | 0 | 1 | 1 | 44 | 0 | 6 | 31 | N.R. | N.R. | 130 | [56] |
| *Cocos nucifera* (CnFatB2) | AEM72523.1 | 2 | 0 | 0 | 1 | 0 | 36 | 1 | 16 | 44 | N.R. | N.R. | 572 | [56] |
| *Cocos nucifera* (CnFatB3) | AEM72524.1 | 11 | 1 | 0 | 34 | 6 | 14 | 23 | 2 | 8 | N.R. | N.R. | 200 | [56] |
| *Cuphea aequipetala* (Ca1FatB1) | AKW88648.1 | 0 | 0 | 0 | 6 | 0 | 51 | 6 | 5 | 32 | 0 | 0 | 140 | [57] |
| *Cuphea aequipetala* (Ca1FatB2) | AKW88636.1 | 95 | 4 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 178 | [57] |
| *Cuphea avigera* (Ca2FatB2) | AKW88670.1 | 40 | 5 | 0 | 2 | 0 | 12 | 2 | 18 | 11 | 3 | 7 | 45 | [57] |
| *Cuphea carthagenesis* (CcFatB) | AKW88645.1 | 1 | 1 | 0 | 20 | 4 | 13 | 24 | 3 | 31 | 1 | 2 | 304 | [57] |
| *Cuphea decandra* (Cd1FatB1) | AKW88663.1 | 0 | 0 | 0 | 0 | 0 | 31 | 1 | 10 | 52 | 0 | 5 | 278 | [57] |
| *Cuphea hookeriana* (ChFatB2)\*\* | AAC49269 | 79 | 21 | N.R. | N.R. | N.R. | N.R. | N.R. | N.R. | N.R. | N.R. | N.R. | 48 | [58],[59] |
| *Cuphea hookeriana* (ChFatB1) | AAC48990.1 | 0 | 0 | 0 | 2 | 0 | 44 | 0 | 38 | 11 | 5 | 0 | 26 | [58],[60] |
| *Cuphea leptopoda* (Cl1FatB1) | AKW88652.1 | 0 | 0 | 0 | 1 | 0 | 44 | 1 | 19 | 25 | 0 | 10 | 182 | [57] |
| *Cuphea leptopoda* (Cl2FatB2) | AKW88658.1 | 3 | 27 | 15 | 5 | 7 | 10 | 2 | 8 | 17 | 0 | 5 | 357 | [57] |
| *Cuphea leptopoda* (Cl3FatB1) | AKW88660.1 | 0 | 0 | 0 | 0 | 0 | 36 | 1 | 13 | 43 | 0 | 6 | 215 | [57] |
| *Cuphea leptopoda* (Cl4FatB1) | AKW88664.1 | 3 | 26 | 12 | 5 | 8 | 11 | 2 | 9 | 18 | 0 | 6 | 249 | [57] |
| *Cuphea palustris* (CpFatB2) | AAC49180.1 | 0 | 1 | 0 | 8 | 0 | 75 | 0 | 16 | 0 | 0 | 0 | 372 | [61] |
| *Cuphea palustris* (CpFatB1) | AAC49179 | 98 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | N.R. | N.R. | 708 | [54] |
| *Cuphea paucipetala* (Cp1FatB1) | AKW88655.1 | 0 | 1 | 0 | 8 | 0 | 75 | 0 | 16 | 0 | 0 | 0 | 372 | [61] |
| *Cuphea viscosissima* (CvFatB1) | AEM72522.1 | 52 | 26 | 1 | 7 | 6 | 7 | 1 | 0 | 0 | N.R. | N.R. | 79 | [54] |
| *Cuphea viscosissima* (CvFatB2) | AEM72523.1 | 4 | 1 | 0 | 1 | 1 | 47 | 1 | 26 | 19 | N.R. | N.R. | 249 | [54] |
| *Cuphea viscosissima* (CvFatB3) | AEM72524.1 | 7 | 5 | 0 | 2 | 2 | 84 | 0 | 0 | 0 | N.R. | N.R. | 19 | [54] |
| *Diploknema butyracea* | AAX51636.1 | 0 | 0 | 0 | 8 | 0 | 7 | 0 | 44 | 5 | 30 | 6 | 151 | [62] |
| *Elaeis guineensis* | AAD42220 | 14 | 2 | 0 | 3 | 3 | 48 | 3 | 0 | 26 | N.R. | N.R. | 37 | [54] |
| *Gossypium hirsutum* | AAD01982.1 | 0 | 0 | 0 | 0 | 0 | 36 | 0 | 21 | 37 | 1 | 5 | 791 | [63] |
| *Helianthus Annuus* (HaFatA1) | AAL79361.1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 44 | 25 | 8 | 23 | 135 | [64] |
| *Helianthus Annuus* (HaFatB1) | AAX19387.1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 36 | 21 | 1 | 41 | N.R. | [65] |
| *Iris germanica 1* | AAG43857 | 3 | 1 | 0 | 1 | 1 | 30 | 0 | 20 | 44 | N.R. | N.R. | 261 | [54] |
| *Iris germanica 2* | AAG43858 | 8 | 0 | 0 | 0 | 0 | 32 | 0 | 23 | 36 | N.R. | N.R. | 15 | [54] |
| *Jatropha curcas* | ABU96744.1 | 0 | 0 | 0 | 0 | 0 | 44 | 0 | 18 | 33 | 1 | 4 | 2130 | [66] |
| *Lindera communis* | AHF72806.1 | 0 | 7 | 0 | 18 | 0 | 17 | 0 | 18 | 21 | 3 | 16 | N.R. | [67] |
| *Micromonas pusilla* | EEH52851 | 4 | 0 | 0 | 0 | 1 | 65 | 8 | 0 | 22 | N.R. | N.R. | 16 | [54] |
| *Physcomitrella patens* | EDQ65090.1 | 9 | 0 | 0 | 0 | 0 | 42 | 0 | 16 | 32 | N.R. | N.R. | 380 | [54] |
| *Prunus sibirica L.* | AIX97815.1 | 0 | 0 | N.R. | 0 | N.R. | 33 | 40 | 0 | 25 | 0 | 0 | 711 | This study |
| *Ricinus communis* | ABV54795.1 | 0 | 0 | 0 | 0 | 0 | 44 | 0 | 17 | 33 | 1 | 4 | 2252 | [66] |
| *Sorghum bicolor 1* | EER87824 | 5 | 0 | 0 | 2 | 0 | 46 | 0 | 13 | 34 | N.R. | N.R. | 126 | [54] |
| *Sorghum bicolor 2* | EER88593 | 6 | 1 | 0 | 3 | 1 | 45 | 3 | 11 | 31 | N.R. | N.R. | 91 | [54] |
| *Sorghum bicolor 4* | EES11622 | 5 | 0 | 0 | 0 | 0 | 51 | 0 | 15 | 29 | N.R. | N.R. | 9 | [54] |
| *Ulmus americana* | AAB71731 | 44 | 23 | 3 | 4 | 9 | 10 | 1 | 1 | 5 | N.R. | N.R. | 1098 | [54],[68] |
| *Umbellularia californica* (UcFatB1) | AAA34215.1 | 44 | 23 | 3 | 4 | 8 | 10 | 2 | 1 | 4 | 4 | 0 | 901 | [69] |
| *Umbellularia californica* (UcFatB2) | AAC49001.1 | 1 | 5 | 0 | 74 | N.R. | 4 | N.R. | 14 | N.R. | 2 | N.R. | 51 | [58] |
| **TE Variants** | **WT GenBank ID** | **% C8** | **% C10** | **% C10:1** | **% C12** | **% C12:1** | **% C14** | **% C14:1** | **% C16** | **% C16:1** | **% C18:0** | **% C18:1** | **mg/L** | **Source** |
| rTE3\* | AEM72523.1 | 45 | 27 | 10 | 4 | 10 | 1 | 2 | 1 | 1 | N.R. | N.R. | 85 | [70] |
| rTE4\* | AEM72523.1 | 26 | 36 | 12 | 4 | 12 | 4 | 4 | 0 | 3 | N.R. | N.R. | 86 | [70] |
| rTE8\* | AEM72523.1 | 23 | 10 | 2 | 3 | 5 | 32 | 6 | 0 | 18 | N.R. | N.R. | 18 | [70] |
| rTE12\* | AEM72523.1 | 22 | 43 | 7 | 3 | 16 | 10 | 0 | 0 | 0 | N.R. | N.R. | 9 | [70] |
| rTE15\* | AEM72523.1 | 8 | 21 | 7 | 15 | 34 | 5 | 10 | 0 | 0 | N.R. | N.R. | 11 | [70] |
| rTE16\* | AEM72523.1 | 41 | 15 | 7 | 9 | 16 | 3 | 8 | 0 | 2 | N.R. | N.R. | 143 | [70] |
| rTE20\* | AEM72523.1 | 17 | 20 | 8 | 12 | 24 | 3 | 13 | 0 | 2 | N.R. | N.R. | 73 | [70] |
| rTE24\* | AEM72523.1 | 22 | 4 | 1 | 6 | 7 | 28 | 6 | 1 | 24 | N.R. | N.R. | 58 | [70] |
| rTE28\* | AEM72523.1 | 6 | 1 | 0 | 1 | 2 | 41 | 2 | 2 | 45 | N.R. | N.R. | 66 | [70] |
| rTE32\* | AEM72523.1 | 27 | 25 | 8 | 11 | 16 | 5 | 4 | 0 | 4 | N.R. | N.R. | 88 | [70] |
| rTE36\* | AEM72523.1 | 8 | 19 | 4 | 13 | 18 | 12 | 17 | 0 | 9 | N.R. | N.R. | 38 | [70] |
| rTE40\* | AEM72523.1 | 23 | 3 | 0 | 6 | 5 | 32 | 11 | 0 | 20 | N.R. | N.R. | 29 | [70] |
| rTE44\* | AEM72523.1 | 13 | 2 | 0 | 1 | 1 | 46 | 6 | 0 | 31 | N.R. | N.R. | 10 | [70] |
| rTE48\* | AEM72523.1 | 39 | 18 | 7 | 9 | 18 | 2 | 6 | 1 | 1 | N.R. | N.R. | 137 | [70] |
| rTE51\* | AEM72523.1 | 6 | 21 | 16 | 2 | 30 | 22 | 2 | 0 | 0 | N.R. | N.R. | 1 | [70] |
| rTE52\* | AEM72523.1 | 56 | 11 | 8 | 5 | 12 | 2 | 5 | 0 | 1 | N.R. | N.R. | 136 | [70] |
| rTE56\* | AEM72523.1 | 30 | 6 | 2 | 8 | 8 | 26 | 5 | 1 | 15 | N.R. | N.R. | 80 | [70] |
| rTE60\* | AEM72523.1 | 4 | 1 | 0 | 1 | 1 | 55 | 0 | 10 | 27 | N.R. | N.R. | 33 | [70] |
| *Cuphea viscosissima*  (CvB2MT17)\* | AEM72523.1 | 30 | 7 | 2 | 18 | 14 | 9 | 9 | 5 | 2 | N.R. | N.R. | 20 | [70] |
| *Cuphea viscosissima*  (CvB2MT6)\* | AEM72523.1 | 46 | 6 | 0 | 22 | 9 | 3 | 14 | 0 | 0 | N.R. | N.R. | 5 | [70] |
| *Cuphea viscosissima*  (CvB2MT34)\* | AEM72523.1 | 42 | 10 | 6 | 9 | 18 | 3 | 11 | 0 | 0 | N.R. | N.R. | 80 | [70] |
| *Cuphea viscosissima*  (CvB2MT33)\* | AEM72523.1 | 36 | 8 | 5 | 10 | 20 | 5 | 13 | 0 | 4 | N.R. | N.R. | 102 | [70] |
| *Cuphea viscosissima*  (CvB2MT25)\* | AEM72523.1 | 37 | 9 | 3 | 13 | 16 | 5 | 14 | 2 | 1 | N.R. | N.R. | 51 | [70] |
| *Cuphea viscosissima*  (CvB2MT47)\* | AEM72523.1 | 20 | 12 | 4 | 18 | 18 | 1 | 5 | 23 | 0 | N.R. | N.R. | 38 | [70] |
| *Cuphea viscosissima*  (CvB2MT45)\* | AEM72523.1 | 26 | 16 | 6 | 13 | 24 | 3 | 11 | 0 | 0 | N.R. | N.R. | 104 | [70] |
| *Cuphea viscosissima*  (CvB2MT44)\* | AEM72523.1 | 27 | 24 | 11 | 8 | 24 | 2 | 4 | 0 | 0 | N.R. | N.R. | 125 | [70] |
| *Cuphea viscosissima*  (CvB2MT42)\* | AEM72523.1 | 36 | 15 | 12 | 7 | 19 | 3 | 6 | 2 | 1 | N.R. | N.R. | 142 | [70] |
| *Cuphea viscosissima*  (CvB2MT20)\* | AEM72523.1 | 33 | 17 | 14 | 7 | 19 | 2 | 6 | 0 | 1 | N.R. | N.R. | 119 | [70] |
| *Cuphea viscosissima*  (CvB2MT30)\* | AEM72523.1 | 42 | 17 | 12 | 6 | 16 | 2 | 5 | 0 | 1 | N.R. | N.R. | 171 | [70] |
| *Cuphea viscosissima*  (CvB2MT48)\* | AEM72523.1 | 15 | 39 | 18 | 7 | 18 | 0 | 4 | 0 | 0 | N.R. | N.R. | 24 | [70] |
| *Cuphea viscosissima*  (CvB2MT40)\* | AEM72523.1 | 68 | 13 | 9 | 3 | 6 | 1 | 1 | 0 | 0 | N.R. | N.R. | 165 | [70] |
| *Cuphea viscosissima*  (CvFatB1)\* | AEM72523.1 | 49 | 26 | 9 | 4 | 10 | 1 | 1 | 0 | 1 | N.R. | N.R. | 88 | [70] |
| *Cuphea viscosissima*  (CvB2MT29)\* | AEM72523.1 | 15 | 7 | 3 | 15 | 18 | 12 | 21 | 0 | 9 | N.R. | N.R. | 58 | [70] |
| *Cuphea viscosissima*  (CvB2MT12)\* | AEM72523.1 | 16 | 7 | 2 | 16 | 15 | 15 | 20 | 0 | 8 | N.R. | N.R. | 30 | [70] |
| *Cuphea viscosissima*  (CvB2MT2)\* | AEM72523.1 | 11 | 5 | 1 | 10 | 10 | 25 | 20 | 0 | 20 | N.R. | N.R. | 18 | [70] |
| *Cuphea viscosissima*  (CvB2MT32)\* | AEM72523.1 | 26 | 7 | 4 | 5 | 14 | 10 | 31 | 0 | 3 | N.R. | N.R. | 64 | [70] |
| *Cuphea viscosissima*  (CvB2MT16)\* | AEM72523.1 | 19 | 6 | 2 | 4 | 10 | 23 | 31 | 0 | 7 | N.R. | N.R. | 63 | [70] |
| *Cuphea viscosissima*  (CvB2MT31)\* | AEM72523.1 | 24 | 6 | 4 | 9 | 20 | 12 | 17 | 1 | 8 | N.R. | N.R. | 76 | [70] |
| *Cuphea viscosissima* (CvB2MT19)\* | AEM72523.1 | 25 | 6 | 3 | 10 | 22 | 9 | 18 | 1 | 6 | N.R. | N.R. | 32 | [70] |
| *Cuphea viscosissima* (CvB2MT15)\* | AEM72523.1 | 22 | 8 | 1 | 16 | 20 | 9 | 20 | 1 | 4 | N.R. | N.R. | 29 | [70] |
| *Cuphea viscosissima* (CvB2MT27)\* | AEM72523.1 | 26 | 10 | 6 | 10 | 22 | 5 | 18 | 0 | 3 | N.R. | N.R. | 43 | [70] |
| *Cuphea viscosissima* (CvB2MT18)\* | AEM72523.1 | 28 | 8 | 3 | 14 | 20 | 6 | 15 | 2 | 3 | N.R. | N.R. | 56 | [70] |
| *Cuphea viscosissima* (CvB2MT22)\* | AEM72523.1 | 25 | 9 | 2 | 13 | 16 | 9 | 17 | 0 | 10 | N.R. | N.R. | 16 | [70] |
| *Cuphea viscosissima* (CvB2MT10)\* | AEM72523.1 | 27 | 10 | 5 | 9 | 14 | 10 | 16 | 0 | 10 | N.R. | N.R. | 33 | [70] |
| *Cuphea viscosissima* (CvB2MT38)\* | AEM72523.1 | 77 | 3 | 1 | 6 | 4 | 2 | 8 | 0 | 0 | N.R. | N.R. | 147 | [70] |
| *Cuphea viscosissima* (CvB2MT35)\* | AEM72523.1 | 77 | 3 | 1 | 4 | 4 | 3 | 9 | 0 | 0 | N.R. | N.R. | 18 | [70] |
| *Cuphea viscosissima* (CvB2MT41)\* | AEM72523.1 | 75 | 4 | 19 | 1 | 1 | 0 | 0 | 0 | 0 | N.R. | N.R. | 13 | [70] |
| *Cuphea viscosissima* (CvB2MT26)\* | AEM72523.1 | 55 | 1 | 0 | 5 | 0 | 18 | 4 | 11 | 6 | N.R. | N.R. | 7 | [70] |
| *Cuphea viscosissima* (CvB2MT36)\* | AEM72523.1 | 98 | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 0 | N.R. | N.R. | 6 | [70] |
| *Cuphea viscosissima* (CvB2MT1)\* | AEM72523.1 | 100 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | N.R. | N.R. | <1 | [70] |
| *Cuphea viscosissima* (CvB2MT37)\* | AEM72523.1 | 89 | 3 | 0 | 3 | 1 | 1 | 2 | 0 | 0 | N.R. | N.R. | 45 | [70] |
| *Cuphea viscosissima* (CvB2MT43)\* | AEM72523.1 | 5 | 1 | 1 | 2 | 1 | 47 | 1 | 12 | 31 | N.R. | N.R. | 41 | [70] |
| *Cuphea viscosissima* (CvB2MT9)\* | AEM72523.1 | 2 | 0 | 0 | 0 | 0 | 53 | 1 | 10 | 34 | N.R. | N.R. | 98 | [70] |
| *Cuphea viscosissima* (CvB2MT14)\* | AEM72523.1 | 4 | 1 | 0 | 2 | 1 | 52 | 1 | 14 | 26 | N.R. | N.R. | 25 | [70] |
| *Cuphea viscosissima* (CvB2MT11)\* | AEM72523.1 | 5 | 2 | 0 | 4 | 4 | 46 | 5 | 12 | 24 | N.R. | N.R. | 28 | [70] |
| *Cuphea viscosissima* (CvB2MT13)\* | AEM72523.1 | 15 | 2 | 1 | 1 | 2 | 51 | 1 | 3 | 26 | N.R. | N.R. | 40 | [70] |
| *Cuphea viscosissima* (CvFatB2)\* | AEM72523.1 | 6 | 5 | 1 | 1 | 3 | 45 | 1 | 6 | 32 | N.R. | N.R. | 40 | [70] |
| *Cuphea viscosissima* (CvB2MT23)\* | AEM72523.1 | 7 | 1 | 0 | 1 | 1 | 54 | 1 | 7 | 29 | N.R. | N.R. | 89 | [70] |
| *Cuphea viscosissima* (CvB2MT8)\* | AEM72523.1 | 5 | 0 | 0 | 0 | 0 | 58 | 0 | 4 | 33 | N.R. | N.R. | 17 | [70] |
| *Cuphea viscosissima* (CvB2MT7)\* | AEM72523.1 | 8 | 1 | 0 | 0 | 0 | 62 | 0 | 1 | 29 | N.R. | N.R. | 11 | [70] |
| *Cuphea viscosissima* (CvB2MT4)\* | AEM72523.1 | 4 | 1 | 0 | 0 | 0 | 61 | 0 | 9 | 25 | N.R. | N.R. | 46 | [70] |
| *Cuphea viscosissima* (CvB2MT24)\* | AEM72523.1 | 4 | 0 | 0 | 0 | 0 | 74 | 0 | 1 | 20 | N.R. | N.R. | 92 | [70] |
| *Cuphea viscosissima* (CvB2MT5)\* | AEM72523.1 | 2 | 0 | 0 | 0 | 0 | 82 | 0 | 3 | 14 | N.R. | N.R. | 106 | [70] |
| *Cuphea viscosissima* (CvB2MT28)\* | AEM72523.1 | 21 | 5 | 1 | 8 | 4 | 40 | 4 | 3 | 14 | N.R. | N.R. | 68 | [70] |
| *Cuphea viscosissima* (CvB2MT21)\* | AEM72523.1 | 21 | 3 | 1 | 5 | 3 | 44 | 5 | 2 | 18 | N.R. | N.R. | 42 | [70] |
| *Cuphea viscosissima* (CvB2MT3)\* | AEM72523.1 | 22 | 4 | 2 | 2 | 5 | 40 | 4 | 0 | 22 | N.R. | N.R. | 19 | [70] |
| UcFatB1(197M, M199H)\*\* | AAA34215.1 | 0 | 0 | 0 | 41 | 0 | 59 | 0 | 0 | 0 | 0 | 0 | 66 | [55] |
| UcFatB1(R197M, M199H, T231K)\*\* | AAA34215.1 | 0 | 0 | 0 | 9 | 0 | 91 | 0 | 0 | 0 | 0 | 0 | 88 | [55] |
| UcFatB1(T231K)\*\* | AAA34215.1 | 1 | 0 | 1 | 14 | 66 | 8 | 3 | 2 | 2 | 4 | 0 | N.R. | [55] |

## Text S4: Molecular biology materials and experimental methods for cloning

## Media and molecular biology materials

Media components were ordered from IBI Scientific (Dubuque, IA) and Fisher Scientific (Waltham, MA). Oligonucleotides were purchased from Integrated DNA Technologies (Coralville, IA) and gene fragments were purchased from Twist Bioscience (San Francisco, CA). Enzymes for DNA cloning were purchased from New England Biolabs (Ipswich, MA).

## Cloning of acyl-ACP TE homologs and variants

TE homologs were tested in the high copy pTRC99a expression vector [71]. Gene fragments were cloned via restriction digestion with *HindIII* and *EcoRI* enzymes and ligated into the pTRC99a construct in place of the BTE gene. Point mutations and truncations were incorporated into the ClFatB3, CwFatB2, and BTE genes by designing mutagenic primers which yielded PCR amplicons amenable to Gibson Assembly [72].

# References

[1] J. Palme, S. Hochreiter, and U. Bodenhofer, “KeBABS: An R package for kernel-based analysis of biological sequences,” *Bioinformatics*, 2015, doi: 10.1093/bioinformatics/btv176.

[2] C. Leslie, E. Eskin, and W. S. Noble, “The spectrum kernel: a string kernel for SVM protein classification.,” *Pac. Symp. Biocomput.*, 2002, doi: 10.1142/9789812799623\_0053.

[3] D. Banerjee, “deeprob/ngrampro: Latest-release.” Jun. 25, 2021, doi: 10.5281/ZENODO.5032503.

[4] D. Jurafsky and J. H. Martin, “Language Modeling with N- grams,” *Speech Lang. Process.*, 2016.

[5] D. Banerjee, “deeprob/ifeatpro: zenodo-sync.” Jun. 25, 2021, doi: 10.5281/ZENODO.5032498.

[6] Z. Chen *et al.*, “IFeature: A Python package and web server for features extraction and selection from protein and peptide sequences,” *Bioinformatics*, 2018, doi: 10.1093/bioinformatics/bty140.

[7] S. Kawashima and M. Kanehisa, “AAindex: Amino acid index database,” *Nucleic Acids Research*. 2000, doi: 10.1093/nar/28.1.374.

[8] D. Banerjee, “deeprob/pssmpro: Latest-release.” Jun. 25, 2021, doi: 10.5281/ZENODO.5032505.

[9] J. Wang *et al.*, “POSSUM: A bioinformatics toolkit for generating numerical sequence feature descriptors based on PSSM profiles,” *Bioinformatics*, 2017, doi: 10.1093/bioinformatics/btx302.

[10] S. F. Altschul *et al.*, “Gapped BLAST and PSI-BLAST: A new generation of protein database search programs,” *Nucleic Acids Research*. 1997, doi: 10.1093/nar/25.17.3389.

[11] B. E. Suzek, H. Huang, P. McGarvey, R. Mazumder, and C. H. Wu, “UniRef: Comprehensive and non-redundant UniProt reference clusters,” *Bioinformatics*, 2007, doi: 10.1093/bioinformatics/btm098.

[12] T. Li, K. Fan, J. Wang, and W. Wang, “Reduction of protein sequence complexity by residue grouping,” *Protein Eng.*, 2003, doi: 10.1093/protein/gzg044.

[13] C. S. Leslie, E. Eskin, A. Cohen, J. Weston, and W. S. Noble, “Mismatch string kernels for discriminative protein classification,” *Bioinformatics*, 2004, doi: 10.1093/bioinformatics/btg431.

[14] P. Kuksa, P. H. Huang, and V. Pavlovic, “A fast, large-scale learning method for protein sequence classification,” 2008.

[15] M. Bhasin and G. P. S. Raghava, “Classification of nuclear receptors based on amino acid composition and dipeptide composition,” *J. Biol. Chem.*, 2004, doi: 10.1074/jbc.M401932200.

[16] K. Chen, L. Kurgan, and M. Rahbari, “Prediction of protein crystallization using collocation of amino acid pairs,” *Biochem. Biophys. Res. Commun.*, 2007, doi: 10.1016/j.bbrc.2007.02.040.

[17] V. Saravanan and N. Gautham, “Harnessing computational biology for exact linear B-cell epitope prediction: A novel amino acid composition-based feature descriptor,” *Omi. A J. Integr. Biol.*, 2015, doi: 10.1089/omi.2015.0095.

[18] T. Y. Lee, Z. Q. Lin, S. J. Hsieh, N. A. Bretaña, and C. T. Lu, “Exploiting maximal dependence decomposition to identify conserved motifs from a group of aligned signal sequences,” *Bioinformatics*, 2011, doi: 10.1093/bioinformatics/btr291.

[19] Z. P. Feng and C. T. Zhang, “Prediction of membrane protein types based on the hydrophobic index of amino acids,” *J. Protein Chem.*, 2000, doi: 10.1023/A:1007091128394.

[20] R. R. Sokal and B. A. Thomson, “Population structure inferred by local spatial autocorrelation: An example from an Amerindian tribal population,” *Am. J. Phys. Anthropol.*, 2006, doi: 10.1002/ajpa.20250.

[21] D. S. Horne, “Prediction of protein helix content from an autocorrelation analysis of sequence hydrophobicities,” *Biopolymers*, 1988, doi: 10.1002/bip.360270308.

[22] C. Z. Cai, L. Y. Han, Z. L. Ji, X. Chen, and Y. Z. Chen, “SVM-Prot: Web-based support vector machine software for functional classification of a protein from its primary sequence,” *Nucleic Acids Res.*, 2003, doi: 10.1093/nar/gkg600.

[23] I. Dubchak, I. Muchnik, S. R. Holbrook, and S. H. Kim, “Prediction of protein folding class using global description of amino acid sequence,” *Proc. Natl. Acad. Sci. U. S. A.*, 1995, doi: 10.1073/pnas.92.19.8700.

[24] L. Y. Han, C. Z. Cai, S. L. Lo, M. C. M. Chung, and Y. Z. Chen, “Prediction of RNA-binding proteins from primary sequence by a support vector machine approach,” *RNA*, 2004, doi: 10.1261/rna.5890304.

[25] J. Shen *et al.*, “Predicting protein-protein interactions based only on sequences information,” *Proc. Natl. Acad. Sci. U. S. A.*, 2007, doi: 10.1073/pnas.0607879104.

[26] K. C. Chou, “Prediction of protein cellular attributes using pseudo-amino acid composition,” *Proteins Struct. Funct. Genet.*, 2001, doi: 10.1002/prot.1035.

[27] T. Liu, X. Zheng, and J. Wang, “Prediction of protein structural class for low-similarity sequences using support vector machine and PSI-BLAST profile,” *Biochimie*, 2010, doi: 10.1016/j.biochi.2010.06.013.

[28] T. Liu, X. Geng, X. Zheng, R. Li, and J. Wang, “Accurate prediction of protein structural class using auto covariance transformation of PSI-BLAST profiles,” *Amino Acids*, 2012, doi: 10.1007/s00726-011-0964-5.

[29] Y. Guo, L. Yu, Z. Wen, and M. Li, “Using support vector machine combined with auto covariance to predict protein-protein interactions from protein sequences,” *Nucleic Acids Res.*, 2008, doi: 10.1093/nar/gkn159.

[30] S. Ding, Y. Li, Z. Shi, and S. Yan, “A protein structural classes prediction method based on predicted secondary structure and PSI-BLAST profile,” *Biochimie*, 2014, doi: 10.1016/j.biochi.2013.09.013.

[31] P. Tao, T. Liu, X. Li, and L. Chen, “Prediction of protein structural class using tri-gram probabilities of position-specific scoring matrix and recursive feature elimination,” *Amino Acids*, 2015, doi: 10.1007/s00726-014-1878-9.

[32] L. Zhang, X. Zhao, and L. Kong, “Predict protein structural class for low-similarity sequences by evolutionary difference information into the general form of Chou[U+05F3]s pseudo amino acid composition,” *J. Theor. Biol.*, 2014, doi: 10.1016/j.jtbi.2014.04.008.

[33] S. Zhang, F. Ye, and X. Yuan, “Using principal component analysis and support vector machine to predict protein structural class for low-similarity sequences via PSSM,” *J. Biomol. Struct. Dyn.*, 2012, doi: 10.1080/07391102.2011.672627.

[34] H. Saini, G. Raicar, S. Lal, A. Dehzangi, S. Imoto, and A. Sharma, “Protein Fold Recognition Using Genetic Algorithm Optimized Voting Scheme and Profile Bigram,” *J. Softw.*, 2016, doi: 10.17706/jsw.11.8.756-767.

[35] J. Zahiri, O. Yaghoubi, M. Mohammad-Noori, R. Ebrahimpour, and A. Masoudi-Nejad, “PPIevo: Protein-protein interaction prediction from PSSM based evolutionary information,” *Genomics*, 2013, doi: 10.1016/j.ygeno.2013.05.006.

[36] K. C. Chou and H. Bin Shen, “MemType-2L: A Web server for predicting membrane proteins and their types by incorporating evolution information through Pse-PSSM,” *Biochem. Biophys. Res. Commun.*, 2007, doi: 10.1016/j.bbrc.2007.06.027.

[37] E. Y. T. Juan, W. J. Li, J. H. Jhang, and C. H. Chiu, “Predicting protein subcellular localizations for gram-negative bacteria using DP-PSSM and support vector machines,” 2009, doi: 10.1109/CISIS.2009.194.

[38] L. Zou, C. Nan, F. Hu, and J. Hancock, “Accurate prediction of bacterial type IV secreted effectors using amino acid composition and PSSM profiles,” *Bioinformatics*, 2013, doi: 10.1093/bioinformatics/btt554.

[39] C. W. Cheng, E. C. Y. Su, J. K. Hwang, T. Y. Sung, and W. L. Hsu, “Predicting RNA-binding sites of proteins using support vector machines and evolutionary information,” 2008, doi: 10.1186/1471-2105-9-S12-S6.

[40] J. C. Jeong, X. Lin, and X. W. Chen, “On position-specific scoring matrix for protein function prediction,” *IEEE/ACM Trans. Comput. Biol. Bioinforma.*, 2011, doi: 10.1109/TCBB.2010.93.

[41] NumPy, “NumPy — NumPy,” *NumPy Website*, 2017. .

[42] F. Pedregosa *et al.*, “Scikit-learn: Machine learning in Python,” *J. Mach. Learn. Res.*, 2011.

[43] J. Wang, H. Zheng, Y. Yang, W. Xiao, and T. Liu, “PredDBP-Stack: Prediction of DNA-Binding Proteins from HMM Profiles using a Stacked Ensemble Method,” *Biomed Res. Int.*, vol. 2020, 2020, doi: 10.1155/2020/7297631.

[44] Y. Xiong, Q. Wang, J. Yang, X. Zhu, and D. Q. Wei, “PredT4SE-stack: Prediction of bacterial type IV secreted effectors from protein sequences using a stacked ensemble method,” *Front. Microbiol.*, vol. 9, no. OCT, 2018, doi: 10.3389/fmicb.2018.02571.

[45] L. Nanni, A. Lumini, and S. Brahnam, “An empirical study of different approaches for protein classification,” *Sci. World J.*, vol. 2014, 2014, doi: 10.1155/2014/236717.

[46] H. L. Dai, “Imbalanced protein data classification using ensemble FTM-SVM,” *IEEE Trans. Nanobioscience*, vol. 14, no. 4, 2015, doi: 10.1109/TNB.2015.2431292.

[47] C. Chen *et al.*, “Improving protein-protein interactions prediction accuracy using XGBoost feature selection and stacked ensemble classifier,” *Comput. Biol. Med.*, vol. 123, 2020, doi: 10.1016/j.compbiomed.2020.103899.

[48] A. Mishra, P. Pokhrel, and M. T. Hoque, “StackDPPred: A stacking based prediction of DNA-binding protein from sequence,” *Bioinformatics*, vol. 35, no. 3, 2019, doi: 10.1093/bioinformatics/bty653.

[49] R. M. Lennen, D. J. Braden, R. M. West, J. A. Dumesic, and B. F. Pfleger, “A process for microbial hydrocarbon synthesis: Overproduction of fatty acids in Escherichia coli and catalytic conversion to alkanes,” *Biotechnol. Bioeng.*, vol. 106, no. 2, 2010, doi: 10.1002/bit.22660.

[50] E. E. Wrenbeck, J. R. Klesmith, J. A. Stapleton, A. Adeniran, K. E. J. Tyo, and T. A. Whitehead, “Plasmid-based one-pot saturation mutagenesis,” *Nat. Methods*, vol. 13, no. 11, pp. 928–930, Nov. 2016, doi: 10.1038/nmeth.4029.

[51] N. J. Hernández Lozada *et al.*, “Highly Active C 8 -Acyl-ACP Thioesterase Variant Isolated by a Synthetic Selection Strategy,” *ACS Synth. Biol.*, vol. 7, no. 9, pp. 2205–2215, 2018, doi: 10.1021/acssynbio.8b00215.

[52] G. Chen *et al.*, “Cloning of Acyl-ACP thioesterase FatA from arachis hypogaea L. and its expression in escherichia coli,” *J. Biomed. Biotechnol.*, vol. 2012, pp. 1–6, 2012, doi: 10.1155/2012/652579.

[53] P. Dörmann, T. A. Voelker, and J. B. Ohlrogge, “Cloning and Expression in Escherichia coli of a Novel Thioesterase from Arabidopsis thaliana Specific for Long-Chain Acyl-Acyl Carrier Proteins,” *Arch. Biochem. Biophys.*, vol. 316, no. 1, pp. 612–618, 1995.

[54] S. S. Jha, J. K. Jha, B. Chattopadhyaya, A. Basu, S. K. Sen, and M. K. Maiti, “Plant Physiology and Biochemistry Cloning and characterization of cDNAs encoding for long-chain saturated acyl-ACP thioesterases from the developing seeds of Brassica juncea,” *Plant Physiol. Biochem.*, vol. 48, no. 6, pp. 476–480, 2010, doi: 10.1016/j.plaphy.2010.02.006.

[55] L. Yuan, T. A. Voelker, and D. J. Hawkins, “Modification of the substrate specificity of an acyl-acyl carrier protein thioesterase by protein engineering,” *Proc. Natl. Acad. Sci. U. S. A.*, 1995, doi: 10.1073/pnas.92.23.10639.

[56] F. Jing, “Characterization of acyl-ACP thioesterases for the purpose of diversifying fatty acid synthesis pathway,” 2013.

[57] P. Gordon Roessler and G. Roy, “ACYL-ACP THOESTERASE GENES AND USES THEREFOR,” 8956834 B2, 2015.

[58] A. Jones, H. M. Davies, and T. A. Voelker, “Palmitoyl-Acyl Carrier Protein ( ACP ) Thioesterase and the Evolutionary Origin of Plant ACyl-ACP Thioesterases,” *Plant Cell*, vol. 7, no. March, pp. 359–371, 1995.

[59] K. Dehesh, P. Edwards, T. Hayes, A. M. Cranmer, and J. Fillatti, “Two novel thioesterases are key determinants of the bimodal distribution of acyl chain length of Cuphea palustris seed oil,” *Plant Physiol.*, vol. 110, no. 1, pp. 203–210, 1996, doi: 10.1104/pp.110.1.203.

[60] Y. Feng *et al.*, “Tuning of acyl-ACP thioesterase activity directed for tailored fatty acid synthesis,” *Appl. Microbiol. Biotechnol.*, vol. 102, no. 7, pp. 3173–3182, 2018, doi: 10.1007/s00253-018-8770-6.

[61] M. Ziesack *et al.*, “Chimeric fatty acyl-acyl carrier protein thioesterases provide mechanistic insight into enzyme specificity and expression,” *Appl. Environ. Microbiol.*, vol. 84, no. 10, pp. 1–12, 2018, doi: 10.1128/AEM.02868-17.

[62] J. K. Jha, M. K. Maiti, A. Bhattacharjee, A. Basu, P. C. Sen, and S. K. Sen, “Cloning and functional expression of an acyl-ACP thioesterase FatB type from Diploknema ( Madhuca ) butyracea seeds in Escherichia coli,” *Plant Physiol. Biochem.*, vol. 44, pp. 645–655, 2006, doi: 10.1016/j.plaphy.2006.09.017.

[63] T. T. Huynh, R. M. Pirtle, and K. D. Chapman, “Expression of a Gossypium hirsutum cDNA encoding a FatB palmitoyl- acyl carrier protein thioesterase in Escherichia coli,” vol. 40, pp. 1–9, 2002.

[64] M. J. Serrano-vega, R. Garces, and E. Martinez-Force, “Cloning , characterization and structural model of a FatA-type thioesterase from sunflower seeds ( Helianthus annuus L .),” *Planta*, vol. 221, pp. 868–880, 2005, doi: 10.1007/s00425-005-1502-z.

[65] J. Serrano-vega, R. Garces, and E. Martınez-force, “Cloning and expression of fatty acids biosynthesis key enzymes from sunflower ( Helianthus annuus L .) in Escherichia coli,” *J. Chromatogr. B*, vol. 786, pp. 221–228, 2003, doi: 10.1016/S1570-0232(02)00767-5.

[66] X. Zhang, M. Li, A. Agrawal, and K. Y. San, “Efficient free fatty acid production in Escherichia coli using plant acyl-ACP thioesterases,” *Metab. Eng.*, vol. 13, no. 6, pp. 713–722, 2011, doi: 10.1016/j.ymben.2011.09.007.

[67] S. Dong, J. Huang, Y. Li, J. Zhang, S. Lin, and Z. Zhang, “Cloning, characterization, and expression analysis of acyl-acyl carrier protein (ACP)-thioesterase B from seeds of Chinese Spicehush (Lindera communis),” *Gene*, vol. 542, no. 1, pp. 16–22, 2014, doi: 10.1016/j.gene.2014.03.028.

[68] T. A. Voelker, A. Jones, A. M. Cranmer, and H. M, “Broad-Range and Binary-Range Acyl-Acyl-Carrier-Protein Thioesterases Suggest an Alternative Mechanism for Medium-Chain Production in Seeds,” pp. 669–677, 1997.

[69] T. A. Voelker and H. M. Davies, “Alteration of the Specificity and Regulation of Fatty Acid Synthesis of Escherichia coli by Expression of a Plant Medium- Chain Acyl-Acyl Carrier Protein Thioesterase,” *J. Bacteriol.*, vol. 176, no. 23, pp. 7320–7327, 1994.

[70] F. Jing, L. Zhao, M. D. Yandeau-Nelson, and B. J. Nikolau, “Two distinct domains contribute to the substrate acyl chain length selectivity of plant acyl-ACP thioesterase,” *Nat. Commun.*, vol. 9, no. 1, p. 860, 2018, doi: 10.1038/s41467-018-03310-z.

[71] E. Amann, B. Ochs, and K. J. Abel, “Tightly regulated tac promoter vectors useful for the expression of unfused and fused proteins in Escherichia coli,” *Gene*, vol. 69, no. 2, 1988, doi: 10.1016/0378-1119(88)90440-4.

[72] D. G. Gibson, L. Young, R. Y. Chuang, J. C. Venter, C. A. Hutchison, and H. O. Smith, “Enzymatic assembly of DNA molecules up to several hundred kilobases,” *Nat. Methods*, vol. 6, no. 5, 2009, doi: 10.1038/nmeth.1318.