Typing Plasmids with Distributed Sequence Representation

Moritz Kaufmann[1], Martin Schüle[2], Theo H. M. Smits[1][0000-0002-1237-235X] and Joël F. Pothier[1][0000-0002-9604-7780]

[1] Environmental Genomics and Systems Biology Research Group, Institute of Natural Resource Sciences, Zurich University of Applied Sciences (ZHAW), Einsiedlerstr. 31, 8820 Wädenswil, Switzerland
[2] Bio-Inspired Modeling & Learning Systems, Institute of Applied Simulation, Zurich University of Applied Sciences (ZHAW), Schloss 1, 8820 Wädenswil, Switzerland

moritz.kaufmann@zhaw.ch, martin.schuele@zhaw.ch, theo.smits@zhaw.ch, joel.pothier@zhaw.ch

Abstract. Multidrug resistant bacteria represent an increasing challenge for medicine. In bacteria, most antibiotic resistances are transmitted by plasmids. Therefore, it is important to study the spread of plasmids in detail in order to initiate possible countermeasures. The classification of plasmids can provide insights into the epidemiology and transmission of plasmid-mediated antibiotic resistance. The previous methods to classify plasmids are replicon typing and MOB typing. Both methods are time consuming and labor-intensive. Therefore, a new approach to plasmid typing was developed, which uses word embeddings and support vector machines (SVM) to simplify plasmid typing. Visualizing the word embeddings with *t*-distributed stochastic neighbor embedding (*t*-SNE) shows that the word embeddings finds distinct structure in the plasmid sequences. The SVM assigned the plasmids in the testing dataset with an average accuracy of 85.9% to the correct MOB type.

Keywords: Plasmid Typing, Word Embedding.

1 Background

1.1 Plasmids

Plasmids are extrachromosomal DNA elements with a characteristic number of copies in the host. Plasmids are found in representatives of all three domains *Archaea, Bacteria* and *Eukarya* [1]. Plasmids encode nonessential but often valuable genes for their host [2]. The plasmids allow genes to be horizontally exchanged via recombination and transposition. Since plasmids can enter new hosts via a variety of mechanisms, they can be regarded as a pool of extrachromosomal DNA that is shared across populations. The acquisition of such genes on plasmids enables the bacteria to react quickly to changing environmental influences, e.g. the presence of antibiotics, which would not be the case

if bacterial fitness were only dependent on *de novo* evolution [3]. Plasmids contain genes that are responsible for initiation and the control of replication. In addition they contain genes that encode a wide variety of phenotypes that help their bacterial hosts to exploit and adapt to their environments [4]. These properties are considered as additional functions and include antibiotic and heavy metal resistance, metabolic properties and pathogenicity factors. Such phenotypes have important consequences for human and animal health, environmental processes and microbial adaptation and evolution [5].

1.2 Plasmid Typing

The classification of plasmids can provide insights into the epidemiology and transmission of plasmid-mediated antibiotic resistance. The previous methods to classify plasmids are replicon typing and MOB typing which use variation in replication loci and relaxase proteins, respectively. Replicons include various loci, none of which are universally present in plasmids [6]. On the other hand, relaxases are thought to occur in all plasmids mobilized by the relaxase-*in-cis* mechanism [7,8]. Nevertheless, the relaxase homology may be distant, even in plasmids of the same MOB type [9]. Recent studies show that the current typing schemes are not able to classify the complete diversity of plasmids [10]. As an example, 11% of the plasmids from the dataset ($n = 2097$) of Orlek et al. [10] could not be replicon-typed or MOB typed.

1.3 Word Embeddings

In natural language processing (NLP) a powerful method to represent language is by learning so-called embeddings. An embedding is a vector representation of a text data token. Commonly the tokens are words, and therefore we refer in our explanations to word embeddings, but the method is not restricted to words. In contrast to word vectors created by one-hot-encoding, which are binary, sparse (mostly made of zeros), high-dimensional (same dimensionality as vocabulary), word embeddings are low-dimensional floating-point vectors. In a good word embedding space synonyms have similar word vectors. Also, distance between word vectors reflect semantic and syntactic distances between those words [11]. A popular training technique to learn word embeddings is Word2Vec [12,13]. Word2vec consists of a two-layer neural network that is trained on the current word and its surrounding context words. The use of context words is inspired by the linguistic concept of *distributional hypothesis*, which states that words that appear in the same context have a similar meaning [14].

1.4 Aim of Study

The aim of this study is to determine whether plasmids can be represented as word embeddings, a method normally used in natural language processing, and subsequently classified by machine learning methods.

2 Methods

2.1 Preparing the Dataset

In order to test the new classification method for plasmids, the database with the original queries of Orlek et al. [15] was downloaded [16]. The database consisted of 2097 fully typed, complete, clinically relevant *Enterobacteriaceae* plasmids from the NCBI database. The data of the nucleotide sequences were loaded with the Biostrings package version 2.36.4 [17] to R version 3.5.1 using RStudio version 1.3.959. The nucleotide sequences were translated to amino acid sequences using the Biostrings package [17] using the standard genetic code. All fuzzy and stop codons automatically translated to X and *, respectively, were removed. To remove outliers, which could influence the training behavior of the machine learning methods, a box plot of the plasmid length was created. All plasmids marked as outliers were removed.

2.2 Embedding Representation

Inspired by NLP word embeddings, we created an embedding representation for amino acid sequences. Following Asgari and Mofrad [18], all amino acid sequences were split into triplets. Then, from one sequence, three sequences were created (see Fig. 2). These triplets are the "words" for which the word embedding is constructed. This is done since the most common techniques to study sequences in bioinformatics involves fixed – length overlapping n-grams [19–21].

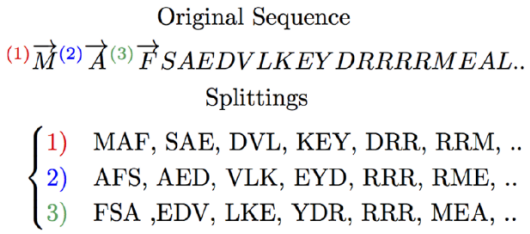

Fig. 1. Schematic illustration of the generated three sequences [18].

The word embeddings were trained using the Skip-Gram algorithm. To calculate the vectors for the embedding the R-package wordVectors version 2.0 was used [22]. The Skip-Gram model learns embeddings by trying to predict context words based on the given target word. Context words are words that occur in a defined window around the target word. Skip-Gram tries to find the corresponding *n*-dimensional vectors for a given training sequence of words, which maximize the log probability function. This gives similar words a similar representation in vector space

$$argmax_{v,y} \frac{1}{N} \sum_{i=1}^{N} \sum_{-c \leq j \leq c, j \neq 0} \log p(w_{i+j}|w_i)$$

$$p(w_{i+j}|w_i) = \frac{\exp(v'^{T}_{w_{i+j}} v_{wi})}{\sum_{k=1}^{W} \exp(v'^{T}_{w_k} v_{wi})} \quad (1)$$

where N is the length of the training sequence, $2c$ is the considered window size for the context, w_i is the center of the window, W is the number of words in the dictionary and v_w and v'_w are input and output n-dimensional representations of word w, respectively. The probability $p(w_{i+j}|w_j)$ is defined by a softmax function. Hierarchical softmax or negative sampling are effective approximations of such a softmax function. The wordVectors package uses negative sampling to approximate the softmax function. Negative sampling uses the following objective function to calculate the word vectors

$$argmax_\theta \prod_{(w,c)\in D} p(D = 1|c, w; \theta) \prod_{(w,c)\in D'} p(D = 0|c, w; \theta) \quad (2)$$

where D is a set of word and context pairs (w, c) existing in the training data set (positive samples) and D' is a randomly generated set of false word and context pairs (w, c) (negative samples). $p(D = 1|w, c; \theta)$ is the probability that (w, c) comes from the training data. $p(D = 0|w, c; \theta)$ is the probability does not come from the training data. The term $p(D = 1|c, w, \theta)$ can be defined as a sigmoid function which can be used for the wordVectors

$$p(D = 1|w, c; \theta) = \frac{1}{1 + e^{-v_c v_w}} \quad (3)$$

Here, the parameters θ are the word vectors we train within the optimization framework v_c while $v_w \in R^d$ are vector representations for the context c and the word w, respectively [23]. In equation 2, the positive samples maximize the probabilities of the observed (w, c) pairs in the training data, while the negative samples prevent all vectors from having the same value by not allowing certain incorrect (w, c) pairs [18]. To train different embeddings different vector sizes and context sizes were chosen. The vocabulary to train the word embeddings were all 8000 possible amino acid triplets. We then represent the entire plasmid as a word embedding, where the amino acid triplets of each reading frame were added for each plasmid. This method follows Asgari and Mofrad [18].

2.3 *t*-Distributed Stochastic Neighbor Embedding (*t*-SNE)

High-dimensional word embeddings can be displayed and interpreted two-dimensionally with the *t*-SNE algorithm. We used the R-package Rtsne version 0.15 [24]. To

evaluate whether the individual MOB types are grouped into clusters, the data points were colored according to the MOB types assigned by Orlek et al. [10]. The *t*-SNE algorithm works as follows: first the similarity score in the original space is calculated from a distance matrix (Euclidean distance) of the input objects

$$p_{j|i} = \frac{exp\left(\frac{-\left|\left|D_{ij}\right|\right|^2}{2\sigma_i^2}\right)}{\sum_{k \neq i} exp\left(\frac{-\left|\left|D_{ij}\right|\right|^2}{2\sigma_i^2}\right)} \tag{4}$$

which is then symmetrized using

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n} \tag{5}$$

The parameter σ of each object is selected so that the perplexity in the original space takes a value as close as possible to the defined perplexity. The perplexity is a parameter that controls how many nearest neighbors are considered when the embedding is generated in low dimensional space. For the low dimensional space, the Cauchy distribution (*t*-distribution with one degree of freedom where the degree of freedom is the number of parameters that may vary independently) is used to represent the distribution of the objects

$$q_{ij} = \frac{\left(1 + \left|\left|y_i - y_j\right|\right|^2\right)^{-1}}{\sum_{k \neq l} \left(1 + ||y_k - y_l||^2\right)^{-1}} \tag{6}$$

The positions of the points in the low dimensional space are determined by minimizing the Kullback-Leiber divergence (KL) of the distribution Q to the distribution P. To minimize the KL-divergence a gradient descent algorithm is used. Since for large datasets a normal gradient descent algorithm would be very computational expensive $O(n^2)$, a Barnes-Hut implementation of the algorithm, is used which leads to a computational complexity of $n\log(n)$. The θ parameter was set to zero to perform an exact *t*-SNE. The *max_iter* parameter was set to 1000. The *PCA* parameter was set to TRUE to perform a PCA prior to the *t*-SNE. To find the best parameters for the perplexity, each model was iterated over 50 cycles. The perplexity parameter was adjusted from 1 to 50. The best fitting perplexity value was chosen according to the lowest KL-divergence.

2.4 Support Vector Machine Classification

To classify the plasmids based on the embedding representation support vector machines are used. SVM with a linear kernel was chosen and implemented with the caret package 6.0-81 [25]. The caret package uses the implementation of the SVM algorithm by kernlab [26]. The SVM algorithm of the kernlab package uses the Sequential Minimal Optimization (SMO) algorithm of Platt [27] to solve the quadratic programming (QP) optimization problem of the SVM. Training an SVM usually requires solving a very big QP optimization problem. The SMO algorithm breaks these big QP optimization problems into a series of smallest possible QP problems. The small QP problems

can then be solved analytically, saving the time consuming numerical solving of a large QP problem [27]. To train the SVM, the data were first centered by subtracting the mean and then scaled by the division of the standard deviation. The partition of the data was 0.8/0.2 for training and test for each iteration. The method for optimizing the tuning parameters was random search.

2.5 BLAST

The BLAST searches to confirm the MOB types of before unclassified plasmids were carried out using the NCBI online tool tblastn version 2.8.1. The algorithm parameters were set to default. The search results were then filtered according to the used thresholds for original MOB type queries used by Orlek et al. [10].

2.6 System

The analyses were run on a PC equipped with an Intel Core i7-3930K processor cadenced at 3.20GHz (6 physical cores, 12 logical cores) and with 64 GB of physical memory.

3 Results

3.1 Data exploration

Unknown plasmids account for around 700 occurrences in the data set with the original queries of Orlek et al [15]. The types MOB_F and MOB_P occur about 450 times each. MOB_Q and MOB_H were already significantly less present with around 150 counts. The types MOB_C and in particular MOB_V were very limited represented, which could lead to classification problems. The dataset was analyzed to check the plasmid lengths and the MOB class distribution (Figure 1). In total, 61 plasmids were marked as outliers. Even though all outliers are most likely plasmids, they were removed from the dataset, as outliers can have a negative effect on the training of the embedding. The average length is shown in the figure as a dashed line. MOB_V plasmids were the shortest in length, while the group of MOB_Q plasmids encompassed mainly short plasmids. The mean lengths of MOB_C, MOB_P and unclassified plasmids were almost comparable. The longest plasmids were in the MOB_F and MOB_H group, with MOB_H plasmids being longer than MOB_F plasmids. Except for the MOB_V plasmids, all plasmids had a variance of about 50 kb in length.

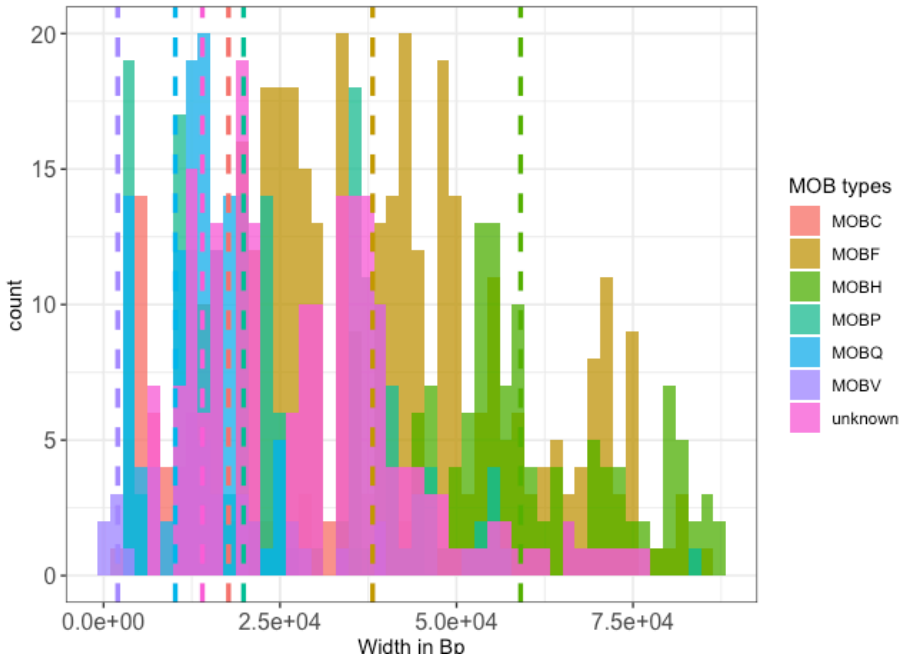

Fig. 3. Distribution of the length of the different MOB types of plasmids. Bp: base pairs.

3.2 *t*-Distributed Stochastic Neighbor Embedding (*t*-SNE)

The lowest KL-divergence was achieved with a perplexity of 49. Figure 3 shows the 1000-dimensional space of the embedding reduced to two dimensions. Each point is a vector representation of a plasmid. A clearly shaped structure was obtained.
For the embeddings with 1000 entries, the SVM assigned the plasmids in the testing dataset with an average accuracy of 85.9% to the correct MOB type. With multi-class classification problems, however, the accuracy does not show the complete picture of the performance of the classifier. The same would apply to a data set with imbalanced classes. Cohen's kappa statistics (κ) is a measure which can handle multi-class and imbalanced classes. For the model, κ was 0.80, indicating that the value is good to excellent according to Greve and Wentura [28] and at the upper end with substantial agreement according to Landis and Koch [29]. Table 1 shows that the classification of MOB_F and MOB_H was very successful with 93.8% and 97.4% balanced accuracy. The confusion matrix also showed that MOB_H was not confused with MOB_F, although the plasmids in Figure 3 were very close to each other. MOB_P was detected with 87.6% accuracy, which is still above the SVM average for all classes. However, MOB_P was confused with MOB_F in 10.7% of the cases. Furthermore, MOB_P was in 3.6% wrongly assigned to type MOB_Q. Orlek et al. [10] reported problems to distinguish between MOB_P and MOB_Q. In the prediction column of MOB_Q, 13 of the total of 51 MOB_Q plasmids were assigned to class MOB_P, which corresponded to 25.5% of all MOB_Q

plasmids. However, this also meant that the representation of the plasmids with embeddings worked well, as the results were congruent with the previously obtained results from Orlek et al. [10]. MOB_Q and MOB_V showed the smallest accuracies, related to an underrepresentation of both classes in the training set. In the testing dataset, only very few plasmids with the respective classes were present and an inconsistent classification has a fatal effect on the accuracy.

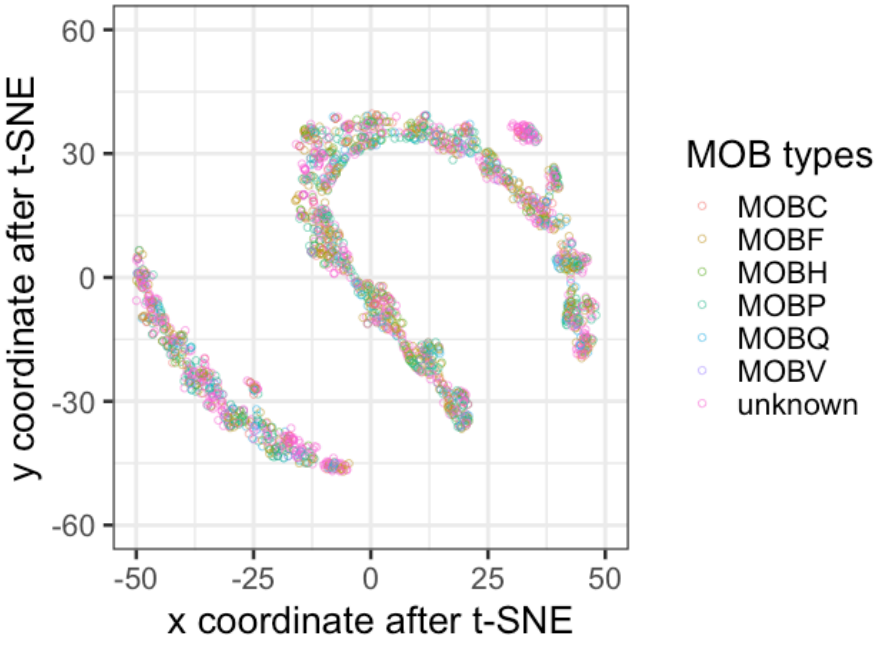

Fig. 4. Two-dimensional space representation of the 1000-dimensional word embeddings of the plasmids.

Table 1. Confusion matrix SVM.

Predictions	Reference						Balanced accuracy
	MOB_C	MOB_F	MOB_H	MOB_P	MOB_Q	MOB_V	
MOB_C	14	2	1	0	2	0	0.812
MOB_F	2	135	0	15	2	0	0.938
MOB_H	0	0	37	0	0	0	0.974
MOB_P	6	5	1	120	13	2	0.876
MOB_Q	0	0	0	5	34	0	0.826
MOB_V	0	0	0	0	0	1	0.667

To investigate whether the SVM can be used to classify plasmids, which were previously unclassifiable by Orlek et al. [10], 939 unclassified plasmids from the data set

were tested with the SVM. The plasmids that could get classified with the SVM were subsequently checked with BLAST against the corresponding proteins used by Orlek et al. [10] for testing. Table 2 shows that 96 plasmids were assigned to a MOB type. Taking into account the thresholds of Orlek et al. [10], 62 plasmids could, after checking with BLAST, still be assigned to a MOB type with relative security. This corresponds to a decrease of unclassifiable plasmids of 3.04%.

Table 2. BLAST verification of the SVM predicted MOB types.

MOB type	Total predicted	Verified with BLAST	Number of predictions < *E*-value threshold	*E*-value threshold
MOB_C	101	42	42	0.001
MOB_F	127	14	8	0.01
MOB_H	17	8	0	0.01
MOB_P	582	27	12	1
MOB_Q	92	2	0	0.0001
MOB_V	20	3	0	0.01
Total	939	96	62	-

4 Discussion

The aim of this work was to represent plasmids as word embeddings and to perform MOB typing using the word embeddings. Asgari and Mofrad [18] were able to classify proteins using word embeddings. However, the method has never been applied to whole plasmids. As could be shown in this work, the word embeddings of entire plasmids can be used to assign the correct MOB types to these plasmids. Based on the available data, MOB typing using SVM seems to be the most successful approach. On the other side, it is possible that with more plasmid sequences present, an approach based on a neural network outperforms the SVM.

By means of the *t*-SNE of the word embeddings, it became clear that the word embeddings represent an up to now not identified structure found in plasmids. The position on the Y-axis could correlate with the length of the plasmids. However, the factor that influenced the position on the X-axis could not be identified. The reconstruction of the plasmid typing, where only the word embeddings of the entire amino acid sequences were used, was functional. In the data set, the accuracy of the test data set was 85.9 %, even though the whole plasmid sequences were only represented by a vector of 1000 entries. Nevertheless, the important factors to assign a MOB type seem to be precisely represented in the word embeddings. As the current version of the SVM was only trained on the known MOB types, one of the currently included MOB type is assigned to each plasmid, since the SVM does not know an unknown type. Nevertheless the MOB type could be set for 62 plasmids, which were before not assigned to any MOB type. These results were then confirmed with a BLAST search.

The classification of the word embeddings is currently based on the biological approach of MOB typing. As long as it is not clear for the already used biological method, which proteins have to be used as queries to get the best results or how many different MOB types exist, the word embedding classification cannot be improved. However, as soon as more biological information about MOB types is available, reconstructing typing with word embeddings offers an interesting alternative. The model only needs to be trained once and can then readily be used. An assignment to a MOB type only takes a fraction of a second and does not require any time-consuming BLAST analysis.

For the next steps it would be conceivable to create a new data set of plasmids. The GenBank database at NCBI continuously includes more plasmids from genome sequencing projects and probably contains a more balanced representation of all plasmid types than at the creation of the used dataset by Orlek et al. [10]. Furthermore, the scripts can be optimized to improve performance. Further tuning of the hyperparameters of the SVM will lead to even better results in the future. It is also conceivable that the MOB typing by word embeddings can be used to establish previously unknown MOB types. As shown in this paper, machine learning methods offer interesting alternatives for conventional bioinformatics approaches and will certainly make their way into biological research soon.

References.

1. Woese, C. R., Kandler, O., Wheelis, M. L.: Towards a natural system of organisms: proposal for the domains Archaea, Bacteria, and Eucarya. Proc. Natl. Acad. Sci. 87, 4576–4579 (1990). doi: 10.1073/pnas.87.12.4576
2. Novick, R. P., Hoppensteadt, F. C.: On plasmid incompatibility. Plasmid 1, 421–434 (1978). doi: 10.1016/0147-619X(78)90001-X
3. Smets, B. F., Barkay, T.: Horizontal gene transfer: perspectives at a crossroads of scientific disciplines. Nat. Rev. Microbiol. 3, 675–678 (2005). doi: 10.1038/nrmicro1253
4. Frost, L. S., Leplae, R., Summers, A. O., Toussaint, A.: Mobile genetic elements: the agents of open source evolution. Nat. Rev. Microbiol. 3, 722–732 (2005). doi: 10.1038/nrmicro1235
5. Johnson, T. J., Nolan, L. K.: Plasmid Replicon Typing. In: Caugant, D. A. (ed.) CEUR Workshop Proceedings, vol. 551, pp. 27–35. Humana Press, Totowa (2009). doi: 10.1007/978-1-60327-999-4_3
6. del Solar, G., Giraldo, R., Ruiz-Echevarría, M. J., Espinosa, M., Díaz-Orejas, R.: Replication and control of circular bacterial plasmids. Microbiol. Mol. Biol. Rev. 62, 434–464 (1998). doi: 1092-2172/98/$04.0010
7. Garcillán-Barcia, M. P., Alvarado, A., de la Cruz, F.: Identification of bacterial plasmids based on mobility and plasmid population biology. FEMS Microbiol. Rev. 35, 936–956 (2011). doi: 10.1111/j.1574-6976.2011.00291.x
8. Ramsay, J. P., Kwong, S. M., Murphy, R. J. T., Yui, E. K., Price, K. J., Nguyen, Q. T., O'Brien, F. G., Grubb, W. B., Coombs, W. B., Firthm N.: An updated view of plasmid conjugation and mobilization in *Staphylococcus*. Mob. Genet. Elements 6, e1208317 (2016). doi: 10.1080/2159256X.2016.1208317
9. Garcillán-Barcia, M. P., Francia, M. V., de La Cruz, F.: The diversity of conjugative

relaxases and its application in plasmid classification. FEMS Microbiol. Rev. 33, 657–687 (2009). doi: 10.1111/j.1574-6976.2009.00168.x

10. Orlek, A., Phan, A., Sheppard, A. E., Doumith, M., Ellington, M., Peto, T., Crook, D., Walker, A. S., Woodford, N., Anjum, M. F., Stoesser, N.: Ordering the mob: Insights into replicon and MOB typing schemes from analysis of a curated dataset of publicly available plasmids. *Plasmid* **91**, 42–52 (2017). doi: 10.1016/j.plasmid.2017.03.002
11. Chollet, F. F., Allaire, J. J.: Deep Learning with R. Manning Publications, Shelter Island (2018).
12. Mikolov, T., Yih, W., Zweig, G.: Linguistic Regularities in Continuous Space Word Representations. In: Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT-2013), pp. 746–751. Association for Computational Linguistics, Atlanta (2013).
13. Brownlee, J.: Word Embeddings. In: Deep Learning for Natural Language Processing, pp. 114–143. Machine Learning Mastery, Vermont Victoria (2017).
14. Harris, Z. S.: Distributional Structure. Word 10, 146–162 (1954). doi: 10.1080/00437956.1954.11659520
15. Orlek, A., Phan, A., Sheppard, A. E., Doumith, M., Ellington, M., Peto, T., Crook, D., Walker, A. S., Woodford, N., Anjum, M. F., Stoesser, N.: A curated dataset of complete Enterobacteriaceae plasmids compiled from the NCBI nucleotide database. Data Br. 12, 423–426 (2017). doi: 10.1016/j.dib.2017.04.024
16. Orlek, A., Phan, A., Sheppard, A. E., Doumith, M., Ellington, M., Peto, T., Crook, D., Walker, A. S., Woodford, N., Anjum, M. F., Stoesser, N.: figshare. https://figshare.com/s/18de8bdcbba47dbaba41 (2017).
17. Pagès, H., Abonyoun, P., Gentleman, R., DebRoy, S.: Biostrings: Efficient manipulation of biological strings. R package version 2.56.0 (2018).
18. Asgari, E., Mofrad, M. R. K.: Continuous Distributed Representation of Biological Sequences for Deep Proteomics and Genomics. PLoS One 10, e0141287 (2015). doi: 10.1371/journal.pone.0141287
19. Ganapathiraju, M., Weisser, D., Rosenfeld, R., Carbonell, P., Reddy, R., Klein-Seetharaman, J.: Comparative N-Gram Analysis of Whole-Genome Protein Sequences. In: Proceedings of the Second International Conference on Human Language Technology Research, pp. 76–81. Morgan Kaufmann, San Francisco (2002).
20. Srinivasan, S. M., Vural, S., King, B. R., Guda, C.: Mining for class-specific motifs in protein sequence classification. BMC Bioinformatics 14, 96 (2013). doi: 10.1186/1471-2105-14-96
21. Vries, J. K., Liu, X.: Subfamily specific conservation profiles for proteins based on n-gram patterns. BMC Bioinformatics 9, 72 (2008). doi: 10.1186/1471-2105-9-72
22. Bmschmidt.: WordVectors. github https://github.com/bmschmidt/wordVectors (2017).
23. Goldenberg, Y., Levy, O.: word2vec Explained: Deriving Mikolov et al.'s Negative-Sampling Word-Embedding Method. In: ArXiv 1402.3722 (2014).
24. Krijthe, J. H.: Rtsne: T-Distributed Stochastic Neighbor Embedding using Barnes-Hut Implementation. https://github.com/jkrijthe/Rtsne (2015).
25. Kuhn, M.: Building Predictive Models in R Using the caret Package. J. Stat. Softw. 28, 1–26 (2008). doi: 10.18637/jss.v028.i05

26. Karatzoglou, A., Smola, A., Zeileis, A.: kernlab – An S4 Package for Kernel Methods in R. J. Stat. Softw. 11, 1–20 (2004).
27. Platt, J. C.: Sequential Minimal Optimization : A Fast Algorithm for Training Support Vector Machines. MSR-TR-98-14 (1998).
28. Greve, W., Wentura, D.: Wissenschaftliche Beobachtung eine Einführung. Beltz, Weinheim (1997).
29. Landis, R., Koch, G.: The Measurement of Observer Agreement for Categorical Data. Biometrics 33, 159–174 (1977).