Graded Vector Representations of Immunoglobulins Produced in Response to West Nile Virus

Trevor Cohen[1], Dominic Widdows[2], Jason A. Vander Heiden[3], Namita T. Gupta[3], and Steven H. Kleinstein[4]

[1] University of Texas School of Biomedical Informatics at Houston
[2] Grab Technologies, Inc.
[3] Interdepartmental Program in Computational Biology and Bioinformatics, Yale University
[4] Departments of Pathology and Immunobiology, Yale School of Medicine

Abstract. Semantic vector models are used to generate high-dimensional vector representations of words from their occurrence statistics across large corpora of electronic text. In these models, the occurrence of a word or number in a particular context is treated as a discrete event, including numerical measurements of continuous properties. In addition, the sequence in which words occur is often ignored. In earlier work we have developed approaches to address these limitations, using graded *demarcator vectors* to represent measured distances in high-dimensional space. This permits incorporation of continuous properties, such as the position of a character within a term or a year of birth, into semantic vector models. In the current paper we extend this work by developing a novel representational approach for protein sequences, in which both the positions and the properties of the amino acid components of protein sequences are represented using graded vectors. Evaluation on a set of around 100,000 immunoglobulin receptor sequences derived from subjects recently infected with West Nile Virus (WNV) suggests that encoding positions and properties using graded vectors increases the similarity between immunoglobulin receptor sequences produced by cells from ancestral lines known to have developed in response to WNV, relative to those from other cell lines.

Keywords: Distributional Semantics, Vector Symbolic Architectures, Binary Spatter Code, Quantum Interactions, Computational Immunology

1 Introduction

The application of quantum-related compositional operators to semantic vector representations — vectors that encode the distribution of terms across large text corpora — has been an active area of inquiry for the Quantum Interactions community since its inception [1]. In these models, the presence of a term in a particular context is considered as a discrete event - the term is either present or absent, though it may be present more than once. However, as we have argued previously [2], a vector space model that aims to provide a holistic account of conceptual representation would also need to represent continuous properties. Consider for example the phrase "has an average high temperature of 106°F in July", which refers to the city of Phoenix, Arizona. A semantic vector representation for Phoenix could take into account that the term "106" had been observed, but this would result in a vector representation that is dissimilar to a similarly constructed vector for the city of Las Vegas, which has an average high temperature of 105°F in the same month. Ideally, these semantic vectors would accommodate continuous values of this sort, resulting in vector representations that are proximal when similar, but not identical, measurements are encountered.

To this end, in our recent work we have developed an approach to represent both discrete events and continuous measurements with semantic vector models, using a quantization technique similar to that employed to model angular momentum in quantum mechanics [3]. This approach was originally developed to encode orthographic similarity between words, such that words with matching characters in proximal positions will have similar vector representations [4]. Subsequently, these methods were extended to the more general case of encoding continuous properties of tabular data [2]. In this paper, we develop these ideas further by encoding both sequence and continuous properties, to generate vector representations of protein sequences.

The paper proceeds as follows. First we provide some context for the current work, in relation to immunology and prior Quantum Interactions contributions. Then we introduce the mathematical structures to-be-employed, and their application for the purpose of encoding the position and properties of amino acids within a sequence. We then proceed to the empirical component of this paper, in which we evaluate the extent to which variants of this approach lead to similar vector representations for collections of immunoglobulin (Ig) receptors expressed by B cell clones from the blood of subjects recently infected with West Nile Virus (WNV).

2 Background

The human immune system is a complex learning network of cells and molecules that are responsible for eliminating infections. Despite its inherent complexity, aspects of the immune system are amenable to computational modeling [5]. B cells are members of the adaptive immune system that recognize foreign organisms and molecules (antigens), using a receptor known as the Ig receptor. Once a B cell recognizes an antigen, it undergoes a process of rapid cell division, mutation and selection that leads to generation of cells with receptor variants having increasing affinity for the antigen. B cells that, through mutation, develop receptors with high affinity for an intruding antigen survive and multiply. Those with receptors having low affinity do not. In this way, the immune system adapts to this antigen by customizing the population of B cells to favor those with Ig receptors that recognize it effectively. Through several experimental steps, the DNA sequence of the Ig receptor's antigen binding region can be determined. This DNA sequence can then be translated into the sequence of component amino acids that, in part, encode the binding affinity of the Ig receptor protein.

The technology is available to accomplish such sequencing quickly and inexpensively [6]. These high-throughput sequencing technologies lead to the generation of large numbers of such sequences, which raises the informatics problem of how best to index, retrieve and analyze them. Historically, sequence comparisons have been conducted with algorithms that determine the minimum cost of pairwise alignment, using variants of string edit distance calculated via dynamic programming approaches [7]. A scalable alternative involves utilizing simpler metrics of comparison such as the hamming distance, which calculates the number of amino acids in common at matched positions, without considering approximate relationships in position [8]. These algorithms consider amino acids as discrete symbols, without representing their chemical properties. Alternatively, the average score of amino acid properties in a particular region may be considered [9]. Such metrics discard information concerning position in the sequence. An algorithm for rapid pairwise comparison that can accommodate variations in sequence and utilize biochemical properties would be a desirable alternative.

3 Mathematical Structures and Methods

3.1 Random Indexing

Random Indexing (RI) is an efficient method of generating semantic vector representations of words [10]. The starting point for RI involves generation of random vectors for the contexts in which terms occur. For example, each document may have a random vector. Random vectors are high-dimensional in nature (dimensionality on the order of 1,000 for real vectors), and are generated by assigning a small number (on the order of 10) of the elements of a zero vector to +1 or -1 at random. On account of the statistical properties of high-dimensional space, such vectors have a high probability of being orthogonal, or close-to-orthogonal to one another. This is reasonably intuitive when considering sparse vectors with a small number of non-zero values. However, it is also the case that randomly-constructed densely populated vectors have a high probability of being far apart in high-dimensions [11], including the binary vectors with an equal probability of one or zero in each dimension that we will use as a fundamental unit of representation in the current experiments [12]. RI utilizes several of the fundamental operators we will apply in the current research. Firstly, RI involves the generation of mutually close-to-orthogonal random vectors as a fundamental representational unit. In the current paper, and in accordance with our previous work, we will refer to such vectors as *elemental vectors*, and the elemental vector for a term will be denoted $E(\mathrm{term})$. We will use the term "semantic vector" and the denotation $S(\mathrm{term})$ to refer to vectors that are generated through superposition of component vectors, and the symbol $+$ to indicate the superposition operation. For example, RI generates semantic term vector representations by superposing (adding) the random vectors representing each of the n contexts this term occurs in, and normalizing the resulting vector.

3.2 Vector Symbolic Architectures

Our approach to encoding sequence and properties draw on a family of representational approaches known as Vector Symbolic Architectures [13] (VSAs). VSAs emerged in response to the critique that connectionist representations, such as neural networks, could not support composition of nested structures thought to underlie reasoning and language, and as such could not provide a comprehensive account of cognition [14]. In connectionist models, the unit of representation is a vector of activation weights. VSAs provide the means to generate compositional structures from such vectors, by providing space-efficient alternatives to Smolensky's initial application of the tensor product for this purpose [15]. These alternatives include circular convolution [16] and element-wise exclusive or (XOR) [17]. In the latter case, the underlying representational unit is a high- (or hyper-) dimensional binary vector, and the VSA is known as the Binary Spatter Code (BSC) [17]. These operators, which are known as *binding* operators, and will be depicted in this paper with the symbol $\otimes$, provide the means for composition. For example, if "temperature" and "106" are represented by vectors, the bound product of these vectors, $\overrightarrow{temperature} \otimes \overrightarrow{106}$, can be used to implement binding of the value "106" to the variable "temperature". A key feature of binding operators is that they are invertible, albeit approximately in some cases. Consequently, we would anticipate if $C = A \otimes B$ then $B \approx C \oslash A$, where $\oslash$ represents the inverse of the binding operator, sometimes called *release*. This mechanism can be employed to retrieve the value bound to a variable, or the representation of the variable to which a particular value is bound. In addition to binding operators, VSAs employ superposition $(+)$ as a compositional operator. Of note, for the current research the symbol $+$ here indicates probabilistic superposition of binary vectors rather than the majority rule that is prescribed by the BSC. The number of 1s and 0s in each dimension are tallied across the component vectors, and the superposition is generated probabilistically, such that $P(x = 1)$ in any dimension is equal to $count(x = 1) \div count(x = 1 \cup x = 0)$.

These operators have different functions: superposition of two vectors produces a vector that is similar to both of its components. Binding produces a vector that is dissimilar from them. The latter property is important for the function of VSAs, as it means that the same value bound to different variables will produce dissimilar vectors: $similarity(\overrightarrow{born} \otimes \overrightarrow{1917}, \overrightarrow{died} \otimes \overrightarrow{1917}) \approx 0$. The use of elemental vectors as a fundamental representational unit entails that values bound to different variables will not be confused with one another: if $E(\text{born}) \approx\perp E(\text{died})$, then $E(\text{born}) \otimes E(1917) \approx\perp E(\text{died}) \otimes E(1917)$. However, relaxing the constraint that these units of representation be mutually close-to-orthogonal provides the means to encode continuous values, such as relative position within a sequence, into semantic vector representations.

3.3 Encoding Sequence

Our approach to encoding sequences was initially developed to encode orthographic representations of words [4]. Consider the word "monk". One way to use VSAs to generate an orthographic representation of this term would be to treat positions as variables, and characters as values [18], exemplifying an approach that is known as "slot coding" [19]. We can then generate an orthographic vector for "monk" as follows:

$$O(monk) = E(\text{m}) \otimes E(1) + E(\text{o}) \otimes E(2) + E(\text{n}) \otimes E(3) + E(\text{k}) \otimes E(4)$$

However, this approach would result in orthographic vector representations that are similar if, and only if, two terms have identical characters in identical positions. This would be inconsistent with the highly flexible nature of human orthographic encoding, which is robust to transposition of characters, amongst other perturbations of sequence. This limitation also applies to VSA-based approaches to encoding sequence that involve the exclusive use of permutations to encode position [20], where a permutation, such as shifting all elements *n* positions to the right, is applied to elemental vectors to indicate their position. A more flexible VSA-based approach to encoding sequence involves the generation of bound products representing n-grams, such as the bigram $E(\mathrm{w}) \otimes E(\mathrm{o})$ [21, 18]. However, the need to encode multiple n-grams and the encoding of "skip-grams" that permit wildcard characters results in a large number of encoding operations for each word - for example, up to fifteen superpositions and sixteen binding operations to generate a bi- and uni-gram based orthographic representation of a five-letter word [22]. In addition, the resulting representations would only be similar for terms with identical bigrams (or skipgrams). Similarity is a measure of the extent to which discrete symbolic representations of character subsequences match one another exactly.

Our approach to encoding sequence is different, in that character positions are treated continuously. This is accomplished by generating a pair of mutually close-to-orthogonal *demarcator* vectors ($D(\mathrm{value})$), which we will call $D(\alpha)$ and $D(\omega)$, and interpolating between them. For example, consider again the word "monk". Given four character positions, p_1, p_2, p_3 and p_4, and a pair of approximately orthogonal demarcator vectors $D(\alpha)$ and $D(\omega)$, we would construct demarcator vectors for these positions such that $D(p_1) = \frac{4}{5}D(\alpha) + \frac{1}{5}D(\omega)$, $D(p_2) = \frac{3}{5}D(\alpha) + \frac{2}{5}D(\omega)$, $D(p_3) = \frac{2}{5}D(\alpha) + \frac{3}{5}D(\omega)$, and $D(p_4) = \frac{1}{5}D(\alpha) + \frac{4}{5}D(\omega)$ [5]. In high dimensions, this results in a set of demarcator vectors in which $sim(D(p_1), D(p_2)) \approx sim(D(p_2), D(p_3)) > sim(D(p_1), D(p_3)) > sim(D(p_1), D(p_4))$ [4]. With these demarcator vectors established an orthographic vector for the word monk, and similarly for any other four-letter word, can be generated as follows:

$$O(monk) = E(\mathrm{m}) \otimes D(\mathrm{p}_1) + E(\mathrm{o}) \otimes D(\mathrm{p}_2) + E(\mathrm{n}) \otimes D(\mathrm{p}_3) + E(\mathrm{k}) \otimes D(\mathrm{p}_4)$$

The resulting representation will be similar to the orthographic vectors for words that have the same characters in similar positions. Better character alignment results in higher similarity. Examples and results are presented in [4], which also discusses the fit between the model and findings from cognitive research on human word recognition.

3.4 Encoding Properties with Graded Values

Subsequently, this approach was generalized as a means to encode continuous values. The procedure in this case involves generating $D(\alpha)$ and $D(\omega)$ for each continuous property of a concept to be represented with a semantic vector. For example, the vector for a city may encode its average temperature, population and square mileage. Then, the minimum ($v_{\min}$) and maximum ($v_{\max}$) of each property is calculated . A vector representing any value $v(c)$ of this continuous property is then generated by interpolation:

$$D(vc) = \frac{v_{\max} - v(c)}{v_{\max} - v_{\min}} D(\alpha) + \frac{v(c) - v_{\min}}{v_{\max} - v_{\min}} D(\omega).$$

For a concept with multiple attributes, an elemental vector is generated for each attribute. A semantic vector for this concept can then be generated by binding the elemental vector for each attribute, $E(A_i)$ to the demarcator vector representing this

[5] With binary vectors, superposition occcurs probabilistically - if $D(\alpha)$ has a 1 as its first element and $D(\omega)$ does not, $D(p_1)$ is generated with a 0.8 probability of a one in this position.

attribute's value $D(V_i)$, and superposing the resulting attribute-value bound product vectors: $S(C) = \sum_{i=1}^{n} E(A_i) \otimes D(V_i)$. Examples and results are presented in [2].

3.5 Quantum Structures

The encoding process utilized for this purpose draws upon a number of mathematical structures that relate to Quantum Theory. The quantization procedure used to generate demarcator vectors is similar to that used for modeling angular momentum in quantum mechanics [3]. The binding operator employed is equivalent to the use of circular convolution in Circular Holographic Reduced Representations, a complex vector based VSA, with phase angles quantized to 0 and π [16]. Circular convolution in turn derives from the tensor product (for a concise account of the relationship between the tensor product and the binding operators employed in different VSAs, see [23]), used in quantum mechanics to represent composite systems. As has been noted previously, superpositions of role-filler bound products (such as $E(\mathrm{m}) \otimes D(\mathrm{p}_1)$) constitute entangled states [24]. Finally, the variant of the hamming distance employed to compare sequence vectors to one another is equivalent to the cosine metric, and as such bears correspondence to the use of projection operators to estimate the probability of observations.

4 Protein Sequences and Amino Acid Properties

In this section, we will describe how we combine our approach to encoding sequence with our approach to encoding continuous values, to generate vector representations of protein sequences. In addition, we employ a permutation operator to enable the algorithm to distinguish between regions of interest. The permutation operator is used in the context of VSAs to dissociate vectors from one another [25]. The general idea is that, once permuted, a high-dimensional vector is highly likely to be close-to-orthogonal to all other vectors in the space - including the vector to which the permutation operator was applied. In the context of modeling sequence, permutation has been used to ensure that words in each position of a sliding window are treated differently [20]. Following this approach, we use a permutation operation in which we shift the bits of a binary vector n positions to the right, with a different n for each region of interest. For computational convenience we perform this operation blockwise, shifting 64-bit blocks to the right rather than individual bits. An overview of our approach to encoding sequence is provided in Figure 1. The enumeration in the list below refers back to the numbers 1-5 in the rounded rectangles in this figure.

1. DNA sequences are segmented into three-character codons, and translated into amino acids in accordance with the genetic code [26].
2. Representations of amino acids are composed from demarcator and elemental vectors representing property value pairs, as described in Section 3.4. Encoded properties are shown in Table 1. In the case of charge, amino acids without charge were given the value of zero, and other values were converted to charge at pH 7.4 using the Hendersen- Hasselbach equation, such that $charge = (1 + 10^{(7.4 - pK[X])})^{-1}$ for $pK[X] \in \{pK[R], pK[H], pK[K]\}$ and $charge = -(1 + 10^{-(7.4 - pK[X])})^{-1}$

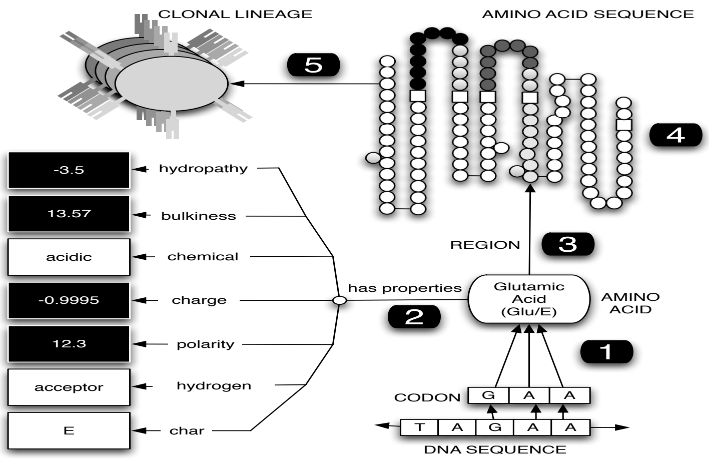

Fig. 1. Overview of encoding processes.

for $pK[X] \in \{pK[D], pK[E], pK[C], pK[Y]\}$ where $pK[X]$ is the negative log of the acid dissociation constant for the amino acid residue X. Categorical values were encoded as the bound product of elemental vectors representing the category and the value concerned. Unmapped codons that do not conform to the genetic code are represented by elemental vectors.

3. Permutation is applied to differentiate regions of a protein sequence from one another. We applied permutations to distinguish between each of the Complementarity Determining Regions (CDR1-3) and Framework Regions (FWR1-3), as defined by the IMGT numbering scheme [27], when encoding structure. These regions were assigned permutations of 1 through 6 64-bit blocks to the right (P^{+1} to P^{+6}).
4. Graded vectors are employed to encode position within a region. For each encoded region, the vector representation of each amino acid is bound to a demarcator vector indicating its position within this region.
5. Vector representations of sequences, composed from region vectors via superposition, are in turn superposed to generate representations of clonal lineages, where a clonal lineage is defined as the population of B cells that are descended from a common ancestor B cell. The vector for a clonal lineage is the superposition of the vectors for all the Ig receptor sequences of that cell population.

5 Evaluation

To evaluate the model, we employed a set of 98,402 Ig sequences derived from three individuals identified as recently infected with the WNV [28]. These sequences originated from 52,505 unique clonal lineages, with clonal lineage membership determined using the Change-O [29] toolkit and the parameters specified in Tsioris and Gupta et al, 2015 [28]. Three of these clones were identified as producing WNV-specific antibodies using a single-cell nanowell approach to identify WNV-specific B cells [28, 30]. We set out to

Table 1. Encoded properties for partial list of amino acids (10 of 20).

	hydropathy	bulkiness	chemical	charge	polarity	hydrogen	char
Ala	1.8	11.5	aliphatic	0	8.1	none	A
Arg	-4.5	14.28	basic	0.9999	10.5	donor	R
Asn	-3.5	12.82	amide	0	11.6	donor/acceptor	N
Asp	-3.5	11.68	acidic	-0.9997	13	acceptor	D
Cys	2.5	13.46	sulfur	-0.0736	5.5	none	C
Gln	-3.5	14.45	amide	0	10.5	donor/acceptor	Q
Glu	-3.5	13.57	acidic	-0.9995	12.3	acceptor	E
Gly	-0.4	3.4	aliphatic	0	9	none	G
His	-3.2	13.69	basic	0.1118	10.4	donor/acceptor	H
Ile	4.5	21.4	aliphatic	0	5.2	none	I

evaluate the extent to which the vector representation of each WNV-specific clonal lineage could serve as a cue to retrieve the remaining two, amongst the 52,504 possibilities (the probability of this occurring by chance is vanishingly small, at $3.8092e^{-5}$).

To evaluate the utility of encoding structure and amino acid properties, we tested several configurations of the model. These are shown in Table 2, and include graded-vector encoding of both structure and property in accordance with the entirety of Figure 1 (GrSP); encoding structure only, without encoding amino acid properties (GrS); ignoring structure and treating the protein sequences as "bags" of either amino acids (BoAA) or amino acid properties (BoP); and "slot coding" approaches in which vector representations of either amino acids (SloAA) or amino acid properties (SloP) are bound to elemental vectors representing their position within the protein sequence, such that the similarity function requires finding the same (SloAA) or a similar (SloP) amino acid, in exactly the same position as that encountered in a cue sequence. In all cases, 52,205 "clone vectors" were generated, each one representing the repertoire of Ig sequences derived from a single clonal lineage.

For each of the model variants shown in Table 2, we generated high-dimensional binary vectors at six different dimensionalities between 1024 and 32,768 ($2^{10}-2^{15}$). For the sake of reproducibility, we generated elemental vectors using a quasi-deterministic approach in which the pseudo-random number generator is seeded with a hash function derived from the term-to-be-represented [31]. This preserves the desirable statistical property of near-orthogonality, while ensuring that the influence of random overlap between elemental vectors is consistent across experiments. For each model, and at each dimensionality, we used the vector representations of each of the three WNV-specific clones as cues. Each cue vector was compared against the other 52,504 clone vectors in the space, which were rank ordered with respect to their similarity, with similarity estimated as $1 - \frac{2}{n}$ hamming distance (HD) [6]. Generation and comparison of high-dimensional binary vectors, including graded vectors, was conducted using the open source Semantic Vectors package [32]. In each case, the ranks of the vectors for the other two WNV-specific clones were recorded.

[6] This corresponds to the cosine metric if binary vectors are treated as vectors in {1,-1} not {1,0}. For example, 1 - (2/4)*HD(1110, 1111) = 0.5, and cos((0.5,0.5,0.5,-0.5) , (0.5,0.5,0.5,0.5)) = 0.5 (with 0.5 for normalized vector components after division by $\sqrt{4}$).

Table 2. Summary of models across 36 cue-by-target-by-dimensionality combinations. Rgn=region encoded via permutation. Pos=position encoded, either with graded vectors (Gr), or with slot coding (Sl).Prop=Properties encoded. AA=Amino Acid.

Model	Rgn	Pos	Prp	Description
GrSP	✓	Gr	✓	"Graded Structure and Properties": property-based AA vectors bound to graded position vectors, with permuted regions.
GrS	✓	Gr		"Graded Structure": elemental AA vectors bound to graded position vectors, with permuted regions.
BoP			✓	"Bag-of-Properties": sum of property-derived AA vectors.
BoAA				"Bag-of-amino-acids": sum of elemental vectors for AAs.
SloP		Sl	✓	"Slotted Properties": BoP + bind to elemental position vectors.
SloAA		Sl		'Slotted Amino Acids": BoAA + bind to elemental position vectors.

6 Results

The results of these experiments are shown in Table 3, which provides the median and minimum rank across the 36 searches (6 dimensionalities x 3 cues x 2 targets), and counts of the number of these examples that fell within the top-ranked results at different thresholds. All of the models evaluted reliably recover WNV-specific clonal lineages within the top 1,000 results (the probability of this occurring at random is around .02). The graded vector based methods (GrSP and GrS) perform best with respect to the number of occurrences of WNV-specific lineages ranked within the top 100 results and above, and the GrSP method has the lowest median rank of retrieval across all cases.

Incorporating information about amino acid properties (grey columns) improves the performance of models that incorporate structure (GrSP > GrS and SloP > SloAA), and vice-versa - GrSP and SloP also outperform BoP with respect to proximal (e.g. recall within top 100) and overall (median rank) performance. However, the effect of incorporating property information when structure is ignored (BoP vs BoAA) is more nuanced, with slightly better recall of higher-ranked results but worse performance overall. These two structure-agnostic models are competitive with respect to their ability to recover WNV-specific lineages in the top 1,000 results, but seldom recover these within the top 100. Of the models that incorporate structure, the graded vector approaches, which accommodate approximate alignment, outperform the tighter constraints of the "slot cod-

Table 3. Summary of results across all examples and dimensionalities. Best results in each row are in **bold**. Models incorporating amino acid properties are in grey columns.

	GrSP	GrS	BoP	BoAA	SloP	SloAA
MEDIAN	**877.5**	3857.5	4980	1416	2940.5	6051.5
MIN	5	**4**	33	98	34	26
<= 10	4	**3**	0	0	0	0
<= 50	**7**	6	1	0	2	1
<= 100	**10**	**10**	1	1	4	1
<= 500	12	14	13	**17**	10	5
<= 1000	**18**	15	17	**18**	11	7

Table 4. Minimum, median and maximum rank for productive cue:target pairs ($\genfrac{}{}{0pt}{}{MIN;MED}{MAX}$). Best MIN in **bold**, best MED underlined, best MAX in *italics* if $\leq$ 1000. Ranks $>$ 1000 in grey text. Models incorporating properties in grey columns. C/T = $\frac{\text{Cue}}{\text{Target}}$.

C/T	GrSP	GrS	BoP	BoAA	SloP	SloAA
125584	**26**; 54	45; 57.5	116; 267.5	107; 227.5	228; 349.5	1632; 5342.5
314052	*111*	9323	337	306	14489	7292
314052	510; 1000	3584; 7608	324; 429	**317**; 472.5	1917; 2296.5	1802; 8434.5
125584	3056	32883	745	*599*	39694	9512
314052	520; 877.5	**96**; 560.5	15230; 20214	10711; 14808	1997; 3199	409; 2208
68974	16851	4131	42451	16032	10117	2842
68974	5; 9	**4**; 12.5	33; 501.5	98; 165	34; 53.5	26; 350.5
314052	*317*	387	8390	418	339	924

ing" approaches. SloAA performs worst by most metrics, which is not surprising as it is the model with the tightest constraints, with insistence upon an exact match between both position and specific amino acid. Table 4 summarizes performance for productive cue:target pairs, excluding 125584:68974 which was not retrieved within the top 1000 results in either orientation. This pattern suggests one target, 314052, is more readily retrieved than others. One explanation for this might be that more Ig sequences related to this clone appear in the data set (n=44 vs n=5 and n=12 for 68974 and 125584 respectively). So the clone vector representing it exhibits a broader range of WNV-related characteristics. Furthermore, superposition of these sequences will emphasize characteristics that are preserved across the ancestral lineage. From a biological perspective, these should be the characteristics that define specificity for WNV.

7 Discussion

In this paper, we develop and evaluate a method through which the position and properties of components of a protein sequence can be encoded into high-dimensional vector representations, such that proteins with similar amino acids in similar positions will have similar vectors. These vectors can be compared efficiently using a variant of the hamming distance. Furthermore, they can be superposed to represent the Ig sequence collection of a particular clonal lineage. Evaluation reveals encoding amino acid properties improves retrieval of one WNV-binding clonal lineage when another is used as a cue, if sequence and structure are encoded also. Encoding sequence and structure improves retrieval with or without encoding these properties, if position is not rigidly encoded. Of note, it is not necessarily the case that clonal lineages ranked higher than the desired targets are *not* WNV-specific. It may prove the case that our approach has identified other sensitized clones - a possibility that would need to be evaluated empirically. Nonetheless, these results suggest our approach may provide a scalable solution to the problem of approximately matching protein sequences, which is a fundamental problem in computational genomics. In our future work, we will continue to develop and evaluate this approach. In particular, we will incorporate a broader range of amino acid properties, and evaluate the approach in the context of larger data sets.

8 Conclusion

In this work, we adapt approaches to encoding sequence and continuous values into semantic vector representations to the task of representing protein sequences. Evaluation suggests that encoding amino acid properties is of value for the identification of proteins with similar immunological specificity, if and only if the position of these amino acids is encoded also. However, it is preferable that this encoding be flexible, permitting approximate match in position. Our approach transforms the computationally demanding task of approximate alignment of sequence into the computationally convenient task of measuring the similarity between semantic vector representations. Consequently, it may be applicable to situations requiring rapid evaluation of large numbers of sequences.

Acknowledgments: This research was supported by NIH/BD2K supplement R01LM011563-S1 and NIH/BD2K supplement R01AI104739-S1.

References

1. S. Clark and S. Pulman, "Combining symbolic and distributional models of meaning.," in *AAAI Spring Symposium: Quantum Interaction*, pp. 52–55, 2007.
2. D. Widdows and T. Cohen, "Graded semantic vectors: An approach to representing graded quantities in generalized quantum models," in *Quantum Interaction*, pp. 231–244, Springer, 2015.
3. D. Bohm, *Quantum Theory*. Prentice-Hall, 1951. Republished by Dover, 1989.
4. T. Cohen, D. Widdows, M. Wahle, and R. Schvaneveldt, "Orthogonality and orthography: introducing measured distance into semantic space," in *Quantum Interaction*, pp. 34–46, Springer, 2013.
5. S. H. Kleinstein, "Getting started in computational immunology," *PLoS Comput Biol*, vol. 4, no. 8, p. e1000128, 2008.
6. J. Benichou, R. Ben-Hamo, Y. Louzoun, and S. Efroni, "Rep-seq: uncovering the immunological repertoire through next-generation sequencing," *Immunology*, vol. 135, no. 3, pp. 183–191, 2012.
7. R. Durbin, S. R. Eddy, A. Krogh, and G. Mitchison, *Biological sequence analysis: probabilistic models of proteins and nucleic acids*. Cambridge university press, 1998.
8. K. Tsioris, N. T. Gupta, A. O. Ogunniyi, R. M. Zimnisky, F. Qian, Y. Yao, X. Wang, J. N. Stern, R. Chari, A. W. Briggs, *et al.*, "Neutralizing antibodies against west nile virus identified directly from human b cells by single-cell analysis and next generation sequencing," *Integrative Biology*, vol. 7, no. 12, pp. 1587–1597, 2015.
9. Y.-C. Wu, D. Kipling, H. S. Leong, V. Martin, A. a. Ademokun, and D. K. Dunn-Walters, "High-throughput immunoglobulin repertoire analysis distinguishes between human IgM memory and switched memory B-cell populations.," *Blood*, vol. 116, pp. 1070–8, aug 2010.
10. P. Kanerva, J. Kristofersson, and A. Holst, "Random indexing of text samples for latent semantic analysis," in *Proceedings of the 22nd Annual Conference of the Cognitive Science Society*, vol. 1036, 2000.
11. D. Widdows and T. Cohen, "Reasoning with vectors: a continuous model for fast robust inference," *Logic Journal of IGPL*, p. jzu028, Nov. 2014.
12. P. Kanerva, *Sparse distributed memory*. Cambridge, Massachusetts: The MIT Press, 1988.
13. R. W. Gayler, "Vector symbolic architectures answer Jackendoff's challenges for cognitive neuroscience," in *In Peter Slezak (Ed.), ICCS/ASCS International Conference on Cognitive Science*, (Sydney, Australia. University of New South Wales.), pp. 133–138, 2004.

14. J. A. Fodor and Z. W. Pylyshyn, “Connectionism and cognitive architecture: A critical analysis,” *Cognition*, vol. 28, no. 1-2, pp. 3–71, 1988.
15. P. Smolensky, “Tensor product variable binding and the representation of symbolic structures in connectionist systems,” *Artificial intelligence*, vol. 46, no. 1, pp. 159–216, 1990.
16. T. A. Plate, *Holographic Reduced Representations: Distributed Representation for Cognitive Structures*. CSLI Publications, 2003.
17. P. Kanerva, “Binary spatter-coding of ordered k-tuples,” *Artificial Neural Networks—ICANN 96*, pp. 869–873, 1996.
18. T. Hannagan, E. Dupoux, and A. Christophe, “Holographic string encoding,” *Cognitive science*, vol. 35, no. 1, pp. 79–118, 2011.
19. C. J. Davis and J. S. Bowers, “Contrasting five different theories of letter position coding: Evidence from orthographic similarity effects.,” *Journal of Experimental Psychology: Human Perception and Performance*, vol. 32, no. 3, p. 535, 2006.
20. M. Sahlgren, A. Holst, and P. Kanerva, “Permutations as a means to encode order in word space.,” in *Proceedings of the 30th Annual Meeting of the Cognitive Science Society (CogSci’08), July 23-26, Washington D.C., USA.*, 2008.
21. M. N. Jones, W. Kintsch, and D. J. Mewhort, “High-dimensional semantic space accounts of priming,” *Journal of memory and language*, vol. 55, no. 4, pp. 534–552, 2006.
22. G. E. Cox, G. Kachergis, G. Recchia, and M. N. Jones, “Toward a scalable holographic word-form representation,” *Behavior research methods*, vol. 43, no. 3, pp. 602–615, 2011.
23. D. Aerts, M. Czachor, and B. De Moor, “Geometric analogue of holographic reduced representation,” *Journal of Mathematical Psychology*, vol. 53, no. 5, pp. 389–398, 2007.
24. D. Aerts and M. Czachor, “Quantum aspects of semantic analysis and symbolic artificial intelligence,” *J. Phys. A: Math. Gen.*, vol. 37, pp. L123–L132, 2004.
25. P. Kanerva, “Hyperdimensional computing: An introduction to computing in distributed representation with high-dimensional random vectors,” *Cognitive Computation*, vol. 1, no. 2, pp. 139–159, 2009.
26. F. Crick, L. Barnett, S. Brenner, and R. J. Watts-Tobin, *General nature of the genetic code for proteins*. Macmillan Journals Limited, 1961.
27. M.-P. Lefranc, C. Pommié, M. Ruiz, V. Giudicelli, E. Foulquier, L. Truong, V. Thouvenin-Contet, and G. Lefranc, “Imgt unique numbering for immunoglobulin and t cell receptor variable domains and ig superfamily v-like domains,” *Developmental & Comparative Immunology*, vol. 27, no. 1, pp. 55–77, 2003.
28. K. Tsioris, N. T. Gupta, A. O. Ogunniyi, R. M. Zimnisky, F. Qian, Y. Yao, X. Wang, J. N. H. Stern, R. Chari, A. W. Briggs, C. R. Clouser, F. Vigneault, G. M. Church, M. N. Garcia, K. O. Murray, R. R. Montgomery, S. H. Kleinstein, and J. C. Love, “Neutralizing antibodies against West Nile virus identified directly from human B cells by single-cell analysis and next generation sequencing,” *Integr. Biol.*, vol. 7, no. 12, pp. 1587–1597, 2015.
29. N. T. Gupta, J. A. Vander Heiden, M. Uduman, D. Gadala-Maria, G. Yaari, and S. H. Kleinstein, “Change-O: a toolkit for analyzing large-scale B cell immunoglobulin repertoire sequencing data: Table 1.,” *Bioinformatics*, vol. 31, pp. 3356–3358, oct 2015.
30. A. O. Ogunniyi, B. A. Thomas, T. J. Politano, N. Varadarajan, E. Landais, P. Poignard, B. D. Walker, D. S. Kwon, and J. C. Love, “Profiling human antibody responses by integrated single-cell analysis,” *Vaccine*, vol. 32, pp. 2866–2873, may 2014.
31. M. Wahle, D. Widdows, J. R. Herskovic, E. V. Bernstam, and T. Cohen, “Deterministic binary vectors for efficient automated indexing of medline/pubmed abstracts,” in *AMIA Annual Symposium Proceedings*, vol. 2012, p. 940, American Medical Informatics Association, 2012.
32. D. Widdows and T. Cohen, “The semantic vectors package: New algorithms and public tools for distributional semantics,” in *Fourth IEEE International Conference on Semantic Computing (ICSC)*, 2010.