Random clique codes

Vincent Gripon, Vitaly Skachek, Warren J. Gross and Michael Rabbat
Department of Electrical and Computer Engineering
McGill University
3480 University St., Montréal, QC H3A 2A7, Canada

***Abstract*—A new family of associative memories based on sparse neural networks has been recently introduced. These memories achieve excellent performance thanks to the use of error-correcting coding principles. In this work, we introduce a new family of codes termed *clique codes*. These codes are based on the cliques in balanced n-partite graphs describing associative memories. In particular, we study an ensemble of random clique codes, and prove that such ensemble contains asymptotically good codes. Furthermore, these codes can be efficiently decoded using the neural networks based associative memories with limited complexity and memory consumption.**

I. Introduction

Recently, a new type of associative memories relying on neural networks has been introduced in [1], [2]. An associative memory is a device that is able to store messages and to retrieve them efficiently given only a part of their content. Contrary to previous models [3], this new device allows to reach nearly optimal efficiency – the ratio of the number of information bits learned to the number of bits used.

The structure of these new memories is inspired by that of modern graph-based codes, such as turbo codes [4] and LDPC codes [5]. More specifically, these memories are based on two new types of specially designed error-correcting codes, one termed *thrifty codes*, and the other one based on cliques in graphs [6].

There are strong similarities between the functioning of an associative memory and that of an error-correcting decoder associated with an erasure channel. Both systems aim at retrieving information messages – or codewords – given a part of their content. On the other hand, an associative memory is designed to learn new messages whereas a decoder does not have any learning abilities.

In this manuscript, we propose to use these neural networks as decoders. We introduce a new family of codes, termed *clique codes*, that are specially designed for efficient decoding using the neural networks architecture. We also prove that some families of these codes based on random graphs have good asymptotic properties. The proposed codes are non-linear, yet they can be concisely described using cliques in n-partite graphs. We note that existing linear codes, which can be efficiently decoded over discrete memoryless channels (DMCs), such as LDPC-like codes, turbo codes or Reed-Solomon codes, are not suitable for efficient decoding by the neural networks due to their algebraic structure.

It was observed by experiment [1] that good performance is achieved in the neural networks associative memory when the fraction of used edges in the corresponding graph is about 20% of the all possible edges. For that reason we consider a random graph model where an edge exists between two vertices with a fixed probability p. Since the overall code is designed to have a fixed rate R, we need to increase the alphabet size of the code when the code length grows.

This paper is structured as follows. Basic notations and definitions are introduced in Section II. The relations between codes and the graph structure of the associative memory are presented in Section III. Maximum-likelihood decoding and random codes are introduced in Section IV. The new random clique codes are proposed and discussed in Section V. The neural networks based decoding algorithm is presented and its performance is analyzed in Section VI. Finally, the discussion is concluded in Section VII.

II. Definitions

Consider a **graph** $\mathcal{G} = (\mathcal{V}, \delta \subseteq \mathcal{V} \times \mathcal{V})$ where $\mathcal{V}$ is the set of vertices and δ is the edge relation. Such a graph is said to be **balanced n-partite**, $n \in \mathbb{N}$, if $\exists (\mathcal{V}_i)_{1 \leq i \leq n} \subseteq \mathcal{V}^n$:

- $\mathcal{V} = \bigsqcup \mathcal{V}_i$ where $\bigsqcup$ is the disjoint union operator,
- $\forall i, j : |\mathcal{V}_i| = |\mathcal{V}_j|$, where $|.|$ denotes the cardinality,
- $\forall i : \forall v_1, v_2 \in \mathcal{V}_i : (v_1, v_2) \notin \delta$.

The sets $\mathcal{V}_i$ are called **clusters** of $\mathcal{G}$.

A **clique** in a graph is a subset of vertices that are fully interconnected. We denote by $[\ell]$ the set of integers between 1 and ℓ. A **word** $\boldsymbol{w}$ over an alphabet $\mathcal{A}$ is a tuple of elements - called symbols - in $\mathcal{A}$. With no loss of generality, we will consider in what follows that alphabet $\mathcal{A}$ is of the form $[\ell]$. A word will be represented by the sequence of its symbols. If $\boldsymbol{w} \in [\ell]^n$, n is called the length of $\boldsymbol{w}$ and it is denoted by $|\boldsymbol{w}|$ (Note that in [1], [2] and [6] the notation c was used rather than n). The i-th coordinate of a word $\boldsymbol{w}$, denoted by w^i, is the i-th symbol of $\boldsymbol{w}$. Given two words of same length $\boldsymbol{w}_1$ and $\boldsymbol{w}_2$, the **Hamming distance** between $\boldsymbol{w}_1$ and $\boldsymbol{w}_2$, denoted $d_H(\boldsymbol{w}_1, \boldsymbol{w}_2)$, is the number of coordinates in $\boldsymbol{w}_1$ and $\boldsymbol{w}_2$ that are not identical (it is 0 if $\boldsymbol{w}_1 = \boldsymbol{w}_2$). A **code** $\mathcal{C}$ of length n over $[\ell]$ is a set of words (called codewords) of length n having symbols in $[\ell]$. Its size is $|\mathcal{C}|$. To simplify presentation of the results in this manuscript, we assume that two codes, which can be obtained one from another using a substitution of symbols, are identical. The **minimum Hamming distance** of the code $\mathcal{C}$, denoted by $d_{min}(\mathcal{C})$, is defined as:

$$d_{min}(\mathcal{C}) = \min_{\substack{\boldsymbol{w}_1, \boldsymbol{w}_2 \in \mathcal{C} \\ \boldsymbol{w}_1 \neq \boldsymbol{w}_2}} d_H(\boldsymbol{w}_1, \boldsymbol{w}_2) \ ,$$

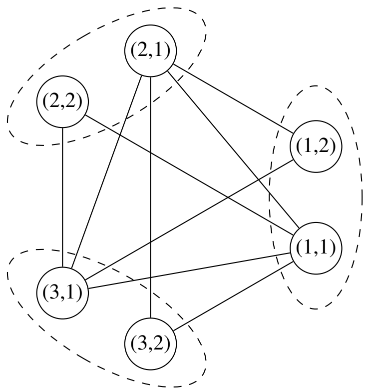

Figure 1. Balanced 3-partite graph ($n = 3$) associated with the code $\mathcal{C} = \{112; 121; 211\}$. For example, this graph contains an edge between nodes (2,2) and (3,1) as the code contains at least one word of the form $\star 21$, where $\star$ denotes any symbol.

and the relative minimum Hamming distance is $\alpha = d_{min}/n$. Finally, we define the **rate** of the code $\mathcal{C}$ as

$$R = \frac{\log_\ell(|\mathcal{C}|)}{n} . \tag{1}$$

III. Clique-closure of codes

Let $\mathcal{C}$ be a code of length $n \in \mathbb{N}$ over $[\ell]$. Such a code can be associated with a balanced n-partite graph using the function $f : \mathcal{C} \mapsto \mathcal{G}$, $\mathcal{G} = (\mathcal{V}, \delta)$ and $\mathcal{V} = [n] \times [\ell]$, such that:

- $\forall i \in [n] \ : \ \mathcal{V}_i = \{i\} \times [\ell]$,
- $\forall (i_1, j_1), (i_2, j_2) \in \mathcal{V} \ : ((i_1, j_1), (i_2, j_2)) \in \delta \Leftrightarrow \exists \boldsymbol{w} \in \mathcal{C} \ : \ w^{i_1} = j_1$ and $w^{i_2} = j_2$.

For example, consider the code $\mathcal{C}_1 = \{112; 121; 211\}$ of length $n = 3$ over [2]. The balanced 3-partite graph associated with $\mathcal{C}_1$ is depicted in Figure 1.

Remark 1. With distinct codes can be associated the same n-partite graph (i.e. f is not injective). For example, consider codes $\mathcal{C}_1$ and $\mathcal{C}_2 = \mathcal{C}_1 \cup \{111\}$.

It can be observed that the set of codes that correspond to a given n-partite graph form a complete partial order according to the relation $\subseteq$ over the sets. This result is stated in Theorem 3. As a prerequisite, we first introduce the maximum code $\mathcal{C}^M$ associated with a balanced n-partite graph.

Definition 2. Consider a balanced n-partite graph $\mathcal{G} = (\mathcal{V}, \delta)$. Let $\mathcal{V}_i = \{(i, 1), (i, 2), \ldots, (i, \ell\}$. The **maximum code** $\mathcal{C}^M$ over $[\ell]$ associated with $\mathcal{G}$ is the code that contains all the words $\boldsymbol{w}$ such that the vertices $\{(i, w^i) \in \mathcal{V} \ : \ 1 \leq i \leq n\}$ form a clique in $\mathcal{G}$.

This leads us to the next theorem.

Theorem 3. *Let $\mathcal{G} = (\mathcal{V}, \delta)$ be a balanced n-partite graph and let $\mathcal{C}^M$ be its associated maximum code. Each code $\mathcal{C}$ such that $f(\mathcal{C}) = \mathcal{G}$ satisfies $\mathcal{C} \subseteq \mathcal{C}^M$.*

Proof: Consider $\mathcal{C}$ such that $f(\mathcal{C}) = \mathcal{G}$ and $\boldsymbol{w} \in \mathcal{C}$. By definition $\forall i, j \in [n], \ i \neq j \ : \ ((i, w^i), (j, w^j)) \in \delta$. Thus, $\{(i, w^i) \ : \ 1 \leq i \leq n\}$ form a clique in $\mathcal{G}$, and so $\boldsymbol{w} \in \mathcal{C}^M$. ∎

The last theorem motivates the following definition.

Definition 4. The **clique-closure** of a code $\mathcal{C}$ is the maximum code associated with $f(\mathcal{C})$.

In the sequel, a code $\mathcal{C}$, whose clique-closure is $\mathcal{C}$ itself, will be called a **clique code**. Its corresponding graph $\mathcal{G}$ will be called a **clique graph**.

IV. ML-decoding

Given an alphabet $\mathcal{A}$, a **memoryless erasure channel** $\mathbb{C}$ associated with $\mathcal{A}$ having erasure probability ρ is a random function $\mathcal{E}^\rho_\mathcal{A} \ : \ \mathcal{A} \to \mathcal{A} \cup \{\perp\}$, where $\perp \notin \mathcal{A}$ is a special "erasure" symbol, such that:

$$\mathcal{E}^\rho_\mathcal{A}(a) = \begin{cases} a & \text{with probability } 1 - \rho \\ \perp & \text{with probability } \rho \ . \end{cases}$$

We extend the definition of $\mathcal{E}^\rho_\mathcal{A}$ to words of length n over $\mathcal{A}$ as follows:

$$\forall \boldsymbol{w} \in \mathcal{A}^n \ : \ \mathcal{E}^\rho_\mathcal{A}(\boldsymbol{w}) = \tilde{\boldsymbol{w}} \text{ such that } \forall i \in [n] \ : \ \tilde{w}^i = \mathcal{E}^\rho_\mathcal{A}(w^i) \ .$$

Assume that the channel $\mathbb{C}$ (or any other DMC) is applied to the *transmitted word* $\boldsymbol{w} \in \mathcal{C}$, and the resulting *received word* $\tilde{\boldsymbol{w}} \in ([\ell] \cup \{\perp\})^n$ is produced. Given $\tilde{\boldsymbol{w}}$, we want to recover $\boldsymbol{w}$. The decoder that minimizes error probability is the *maximum likelihood* (ML) decoder, which is the mapping $D_{ML} \ : \ ([\ell] \cup \{\perp\})^n \to \mathcal{C}$ such that $D_{ML}(\tilde{\boldsymbol{w}}) = \boldsymbol{w}$ whenever

$$\begin{aligned} \forall \boldsymbol{w}' \in \mathcal{C}, \ \boldsymbol{w}' \neq \boldsymbol{w} \ : \ & \mathbb{P}(\tilde{\boldsymbol{w}} \text{ received } | \boldsymbol{w} \text{ transmitted }) \\ & \geq \mathbb{P}(\tilde{\boldsymbol{w}} \text{ received } | \boldsymbol{w}' \text{ transmitted }) \ , \end{aligned} \tag{2}$$

where $\mathbb{P}(\ \cdot \ | \ \cdot \ )$ denotes the conditional probability. For the given channel $\mathbb{C}$, the ML decoder is equivalent to the minimum distance (MD) decoder D_{MD}, which is defined as follows: given the received word $\tilde{\boldsymbol{w}}$,

$$D_{MD}(\tilde{\boldsymbol{w}}) = \arg\min_{\boldsymbol{w} \in \mathcal{C}} d_H(\tilde{\boldsymbol{w}}, \boldsymbol{w}) \ .$$

Let us define the weight distribution function $W \ : \ \mathbb{N} \times \mathcal{C} \to \mathbb{N}$ that associates with a distance $d \in \mathbb{N}$ and a codeword $\boldsymbol{w} \in \mathcal{C}$ the number of codewords at distance d from $\boldsymbol{w}$. A code is said to be identically distributed if

$$\forall \boldsymbol{w}_1, \boldsymbol{w}_2 \in \mathcal{C}, \ \forall d \in \mathbb{N} \ : \ W(d, \boldsymbol{w}_1) = W(d, \boldsymbol{w}_2) = W(d) \ ,$$

i.e. the weight distribution is independent of the codeword.

Typically, for applications one is interested in codes with simultaneously high rate and low decoding error probability. A well-known family of good codes is a random code family. A (nonlinear) random code $\mathcal{C}_r$ of length n over $[\ell]$ can be obtained, for example, by generating all its codewords in the following manner. Each symbol in it is selected randomly and independently, and identically distributed over the elements of $[\ell]$. For such code, the expected weight distribution function is:

$$\mathsf{E}[W(d)] = |\mathcal{C}_r| \cdot \binom{n}{d} \left(\frac{1}{\ell}\right)^{n-d} \left(\frac{\ell - 1}{\ell}\right)^d \ .$$

Generally, in order to admit good decoding performance, a typical codeword should have a small number of codewords at small distances from it.

When using neural networks based decoders, popular algebraic linear codes, such as LDPC-type codes, turbo codes or Reed-Solomon codes, seem to be not suitable for efficient decoding due to their complex algebraic structure, yet they can be decoded by inefficient brute-force methods. However, in general the problem of ML decoding of a arbitrary (linear) code over binary symmetric channel is known to be **NP-hard** [7]. We can think of the naïve ML decoder, which passes through all the codewords until it finds the one that has the best match with the input. This decoder is very inefficient, and its complexity is $\mathbf{C}_{ML} = O(|\mathcal{C}| \cdot n)$. If no compression is used, this algorithm requires a lot of memory: $|\mathcal{C}| \cdot n \log_2(\ell)$ bits (in order to store all the codewords). In the following section, we introduce an alternative family of random nonlinear codes that have rather good minimum distance and can be decoded using an efficient neural networks based algorithm when used over the erasure channel.

V. Random clique codes

Consider an Erdős–Rényi random clique graph [8] with parameter p. In its n-partite graph the edges are selected (given that they are compatible with its n-partite structure) with probability p independently of each other. We call such a graph a **random clique graph** and the maximum code associated with it a **random clique code**.

Theorem 5. *The expected number of codewords in a random clique code is:*

$$p^{\binom{n}{2}} \cdot \ell^n \, . \tag{3}$$

Proof: Consider n vertices from different clusters in $\mathcal{G}$.The probability that those vertices form a clique in $\mathcal{G}$ is the probability that the $\binom{n}{2}$ corresponding edges are in $\mathcal{G}$. This probability is $p^{\binom{n}{2}}$.

Furthermore, the graph contains ℓ^n such sets of vertices, each set forming a distinct possible clique. ∎

Let us now estimate the weight distribution of a random clique code. Given a codeword $\boldsymbol{w}$, the probability that any word $\boldsymbol{w}'$ at distance d from $\boldsymbol{w}$, $d > 0$, is also a codeword is:

$$\left(p^{n-d}\right)^d \cdot p^{\binom{d}{2}} \, .$$

This expression represents the conditional probability that the vertices associated with $\boldsymbol{w}'$ in $\mathcal{G}$ form a clique, given that the edges associated with $\boldsymbol{w}$ form a clique.

Furthermore, the number of possible words at distance d from $\boldsymbol{w}$ is $\binom{n}{d}(\ell-1)^d$. If we assume that ℓ is large, then the edges corresponding to distinct sets of vertices associated with the possible words $\boldsymbol{w}'$ can be considered to be independent.

Lemma 6. Let $\boldsymbol{w}$ be a codeword and suppose that ℓ is large. The expected number of codewords at distance d from $\boldsymbol{w}$ is then:

$$\binom{n}{d}(\ell-1)^d \cdot \left(p^{n-d}\right)^d \cdot p^{\binom{d}{2}} \, . \tag{4}$$

Note that the probability that $\boldsymbol{w}$ is such that their exists another codeword $\boldsymbol{w}'$ at distance d is strictly smaller than the expression given in (4).

Consider a family of random clique codes $\mathcal{F}_p$, where the corresponding random clique graphs are constructed by taking each edge with fixed probability $p > 0$. We have the following lemma:

Lemma 7. If the codes in $\mathcal{F}_p$ are of average rate $\overline{R}$, then ℓ and $\overline{R}$ are connected as follows:

$$\ell = \left(\frac{1}{p}\right)^{\frac{n-1}{2(1-\overline{R})}} \, . \tag{5}$$

Proof: By using (1) and (3) we obtain that the average size of the code $\mathcal{C}$ is $\ell^{\overline{R}n} = p^{\binom{n}{2}} \cdot \ell^n$. By rearranging the terms, we obtain the desired connection. ∎

By combining the above results, we have the following theorem:

Theorem 8. *Let $\overline{d} = \overline{\alpha} n$ be the average minimum Hamming distance of the family of codes $\mathcal{F}_p$, and let $\overline{R}$ be its average rate. Then, for any fixed small $\epsilon > 0$, n and ℓ can be chosen large enough such that:*

$$\overline{R} \geq \frac{\overline{\alpha}(1-\overline{\alpha})}{1 + 2\overline{\alpha} - \overline{\alpha}^2} - \epsilon \, .$$

Proof: Pick a codeword $\boldsymbol{w}$ in a random clique code $\mathcal{C}$ and fix a distance $d > 0$. Denote by $\mathbb{P}_d(\boldsymbol{w})$ the probability that there exists at least one codeword in $\mathcal{C}$ at distance d from $\boldsymbol{w}$. By the union bound, the probability $\hat{\mathbb{P}}_d$ that there exists one pair of codewords in $\mathcal{C}$ at distance d from each other is:

$$\begin{aligned}
\hat{\mathbb{P}}_d &\leq \sum_{\boldsymbol{w} \in \mathcal{C}} \mathbb{P}_d(\boldsymbol{w}) \\
&= \sum_{\boldsymbol{w} \in \mathcal{C}} \binom{n}{d}(\ell-1)^d \cdot \left(p^{n-d}\right)^d \cdot p^{\binom{d}{2}} \\
&= \underbrace{\ell^{n\overline{R}}}_{p_1} \underbrace{\binom{n}{d}}_{p_2} \underbrace{(\ell-1)^d}_{p_3} \left(p^{n-d}\right)^d \cdot p^{\binom{d}{2}} \, .
\end{aligned}$$

Now, by using (5) we express p_1 and p_3 as functions of p. We also approximate p_2 by using the Stirling's approximation ($h_2(\cdot)$ is the binary entropy function). Denote $\alpha = \frac{d}{n}$. Then $\log(\hat{\mathbb{P}}_d)$ can be approximately upper-bounded by:

$$\begin{aligned}
\log(\hat{\mathbb{P}}_d) \leq\ & \log(p)\left[\frac{n(n-1)(\overline{R}+\alpha)}{2(\overline{R}-1)} + \alpha n^2 - \frac{\alpha^2 n^2}{2} - \frac{\alpha n}{2}\right] \\
& + \log(2)(n \cdot h_2(\alpha)) \, .
\end{aligned}$$

When n approaches infinity, this expression is asymptotically equivalent to

$$\log(p) \cdot n^2 \left[\frac{\overline{R}+\alpha}{2(\overline{R}-1)} + \alpha - \frac{\alpha^2}{2}\right] \, . \tag{6}$$

Finally, to force $\hat{\mathbb{P}}_d$ to converge to 0, we require:

$$\frac{\overline{R}+\alpha}{2(\overline{R}-1)} + \alpha - \frac{\alpha^2}{2} < 0 \, .$$

This brings us to the main result of this section:

Theorem 9. *For any fixed value of p, $0 < p < 1$, and for sufficiently large $n \in \mathbb{N}$ and $\ell \in \mathbb{N}$ (where ℓ is appropriately selected based on p and n), there exists a clique code $\mathcal{C}$ with relative minimum Hamming distance at least α and rate R such that*

$$R \geq \frac{\alpha(1-\alpha)}{1+2\alpha-\alpha^2} - \epsilon \ ,$$

for any given small $\epsilon > 0$.

Proof: Theorem 8 connects the average rate and the average relative minimum distance of a random code in $\mathcal{F}$. To obtain the result for a particular code, fix some values of the parameters n and ℓ, and assume that $\mathcal{F}_p$ contains codes $\mathcal{C}_1, \mathcal{C}_2, \cdots, \mathcal{C}_M$ having these values of n and ℓ. Denote by R_i and α_i the rate and the relative minimum Hamming distance, respectively, of $\mathcal{C}_i$.

To arrive at a contradiction, assume that:

$$\forall i \in [M] \ : \ R_i < \frac{\alpha_i(1-\alpha_i)}{1+2\alpha_i-\alpha_i^2} - \epsilon_0$$

for some fixed $\epsilon_0 > 0$. Taking the average over all codes, we obtain:

$$\frac{1}{M}\sum_{i=1}^{M} R_i < \frac{1}{M}\sum_{i=1}^{M} f(\alpha_i) - \epsilon_0 \ .$$

where

$$f(x) = \frac{x(1-x)}{1+2x-x^2}$$

It is easy to check that f is strictly convex over $[0,1]$. Therefore, we obtain

$$\begin{aligned}\overline{R} &= \frac{1}{M}\sum_{i=1}^{M} R_i < \frac{1}{M}\sum_{i=1}^{M} \frac{\alpha_i(1-\alpha_i)}{1+2\alpha_i-\alpha_i^2} - \epsilon_0 \\ &< \frac{\overline{\alpha}(1-\overline{\alpha})}{1+2\overline{\alpha}-\overline{\alpha}^2} - \epsilon_0 \ ,\end{aligned}$$

where the last inequality is due to convexity of $f(\cdot)$. We obtained a contradiction with Theorem 8, and the proof follows. ∎

VI. Decoding clique-closed codes using neural networks

Consider a clique-closed code $\mathcal{C}$. In this section, we show that by substituting vertices with neurons and by providing $\mathcal{G}$ with dynamics, one obtains a neural network that can be used to efficiently decode clique-closed codes.

Assume that the neurons in $\mathcal{V}_i$ are labeled from 1 to ℓ, and denote by $v_{i,j}^t$ the value of neuron j in cluster i at time t, $t \in \mathbb{N}$. In the sequel, we introduce Δ to be the indicator function associated with δ.

Assume that a codeword $\boldsymbol{w}$ was transmitted over an erasure channel $\mathbb{C}$, and the word $\tilde{\boldsymbol{w}}$ was received. This word can be projected onto the neural network as:

$$\forall i \in [n], \ j \in [\ell] \ : \ v_{i,j}^0 = \begin{cases} 1 & \text{if } \tilde{w}^i = j \text{ or } \tilde{w}^i = \perp \\ 0 & \text{otherwise .}\end{cases}$$

	ML-decoder	Neural networks based decoder
Complexity	$O(\ell^{Rn}n)$	$O(n^2\ell^2)$
Memory (bits)	$\ell^{Rn}\log_2(\ell)n$	$\binom{n}{2}\ell^2$

Table I
Comparison of the complexity and of the memory requirements by the ML-decoding algorithm and the efficient neural networks based algorithm.

The following dynamics can be used to decode $\boldsymbol{w}$ from $\tilde{\boldsymbol{w}}$:

$$v_{i,j}^{t+1} = \begin{cases} 1 & \text{if } v_{i,j}^t + \displaystyle\sum_{i'=1}^{n} \max_{1\leq j' \leq l} v_{i',j'}^t \Delta((i,j),(i',j')) \\ & \qquad = n \\ 0 & \text{otherwise}\end{cases} \quad (7)$$

Informally, the dynamics of the network rely on the following principle: all the neurons in the network that are connected to at least one neuron with a non-zero value in each cluster (with the exception of the clusters they are in) are set to 1. The others are set to 0.

Note that this algorithm, if implemented in a naive way and considering a limited number of iterations (in practice a few are enough), has complexity $\mathbf{C} = O(n^2\ell^2)$. Considering a random clique code, this quadratic complexity has to be compared to the exponential complexity of the ML-decoder: $\mathbf{C}_{ML} = O(\ell^{Rn}n)$. In terms of required memory, when assuming that no compression is used, the new algorithm requires $\binom{n}{2}\ell^2$ bits, while the ML-decoder requires $l^{Rn}\log_2(\ell)n$ bits. Table I compares the parameters of the proposed decoding algorithm with that of the ML-decoder.

The proposed decoding algorithm satisfies two properties, stated in Theorem 10 and Theorem 11.

Theorem 10. *The values of neurons in the network always converge.*

Proof: As neuron values are binary, there is a one-to-one bijection between the values of neurons v^t and the set of neurons active at time t: $V_t = \{(i,j) \ : \ v_{i,j} = 1\}$. Therefore it is equivalent to prove that the set of active neurons converges.

V_t is non-increasing with t. As it takes values in a finite space ($\subseteq V_0$), V_t eventually reaches a minimum subset of neurons. ∎

Let us denote by t_∞ an iteration such that convergence is met ($v^{t_\infty} = v^{t_\infty+1}$). In addition, the neurons corresponding to the original message $\boldsymbol{w}$ are active at iteration t_∞:

Theorem 11. $\forall i \ : \ v_{i,w^i}^{t_\infty} = 1$.

Proof: We proceed by induction. To begin, we prove that if $\forall i : v_{i,w^i}^t = 1$ then $\forall i : v_{i,w^i}^{t+1} = 1$. This follows directly from Equation (7), as for each $i \neq j$, $((i,w^i),(j,w^j)) \in \delta$ by construction of $\mathcal{G}$ and $v_{j,w^j}^t = 1$ by hypothesis. Moreover $\forall i : v_{i,w^i}^0 = 1$ by definition. ∎

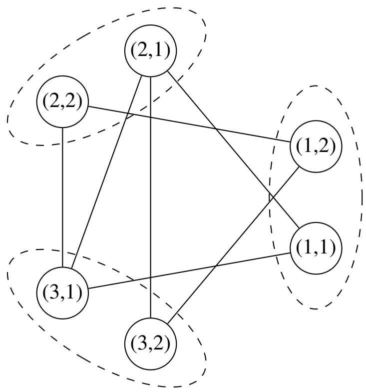

Figure 2. Example of a 3-partite graph containing only one maximum clique and such that using it as a neural decoder results in an ambiguity when decoding the word $(\perp\perp\perp)$.

Observe that decoding using a neural network can result in an ambiguity (several neurons can be active in the same cluster) even if there is only one codeword corresponding to the input. Figure 2 depicts an example of a graph that contains only one maximum clique (corresponding to a code with a single codeword), but such that decoding the entirely erased word $(\perp\perp\perp)$ fails in retrieving the unique codeword. Indeed, each neuron in the network is connected to at least one neuron in the two other clusters. Thus the decoding converges at the first step and keeps all its neurons activated. On the other hand, an ML-decoding would result in (correctly) choosing the codeword (111).

Figure 3 depicts the dependence of the decoding frame error rate as a function of the probability of erasure ρ. Two distinct set of parameters are used. The first set is used for random clique graphs containing $n = 8$ clusters, $\ell = 1000$ vertices in each cluster, with edge probability $p = 0.2$. The second set is used for random clique graphs containing $n = 12$ clusters, $\ell = 100$ vertices each, with edge probability $p = 0.5$. We have chosen families of rather small codes (of rates 0.18 and 0.17, respecitvely) in order to be able to compare them with the ML decoder. For each set of parameters, and each value of ρ, the results are averaged over large number of random graphs and large number of error patterns. The results presented in Figure 3 compare the performance of the neural networks decoding with that of the ML decoding.

VII. Conclusions

We introduced a new family of codes suitable for use with neural networks: random clique codes. These codes are shown to be asymptotically good (as shown in Theorem 9). Furthermore, they can be efficiently decoded using the neural networks based algorithm described in Section VI.

The presented codes are based on the presence of cliques in balanced n-partite graphs. By varying the values of the parameters n, ℓ and p, the codes can be designed to target a specific rate and a specific minimum Hamming distance.

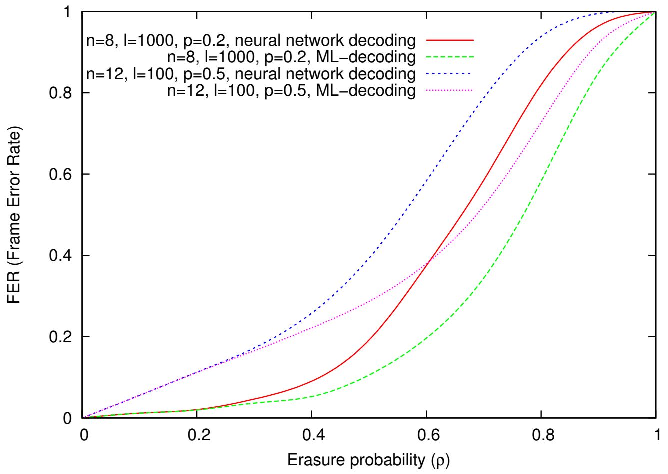

Figure 3. Evolution of the codeword decoding error probability as a function of the erasure channel parameter ρ for an ML-decoder and for the proposed neural networks based decoder.

Many related questions remain open. We have studied only one particular family of random clique codes. Alternative families of random or explicit codes are of interest too. The presented codes are neither linear nor systematic. It would be interesting to obtain more structured constructions of such codes, which admit more efficient encoding and decoding.

Acknowledgment

The authors would like to thank Claude Berrou for providing valuable comments that improved the manuscript.

References

[1] V. Gripon and C. Berrou, “Sparse neural networks with large learning diversity,” *IEEE Transactions on Neural Networks*, vol. 22, pp. 1087 – 1096, Jul. 2011.

[2] V. Gripon and C. Berrou, “A simple and efficient way to store many messages using neural cliques,” *Proceedings of IEEE Symposium on Computational Intelligence, Cognitive Algorithms, Mind, and Brain*, pp. 54 – 58, Apr. 2011.

[3] R. J. McEliece, E. C. Posner, E. R. Rodemich, and S. S. Venkatesh, “The capacity of the Hopfield associative memory,” *IEEE Transactions on Information Theory*, vol. 33, no. 4, pp. 461–482, 1987.

[4] C. Berrou, A. Glavieux, and P. Thitimajshima, “Near Shannon limit error-correcting coding and decoding: Turbo-codes,” *Proceedings of IEEE ICC ’93*, vol. 2, pp. 1064–1070, May 1993.

[5] D. J. MacKay and R. M. Neal, “Good codes based on very sparse matrices,” *Cryptography and Coding. 5th IMA Conference, number 1025 in Lecture Notes in Computer Science*, pp. 100–111, 1995.

[6] V. Gripon and C. Berrou, “Nearly-optimal associative memories based on distributed constant weight codes,” pp. 269–273, February 2012.

[7] E. R. Berlekamp, R. J. McEliece, and H. C. A. Van Tilborg, “On the inherent intractability of certain coding problems,” *IEEE Transactions on Information Theory*, vol. 24, pp. 384–386, 1978.

[8] P. Erdös and A. Rényi, “On random graphs, I,” *Publicationes Mathematicae (Debrecen)*, vol. 6, pp. 290–297, 1959.