# Pointwise HSIC: A Linear-Time Kernelized Co-occurrence Norm for Sparse Linguistic Expressions

**Sho Yokoi** [1,2] **Sosuke Kobayashi** [3] **Kenji Fukumizu** [4] **Jun Suzuki** [1,2] **Kentaro Inui** [1,2]

[1] Tohoku University [2] RIKEN Center for Advanced Intelligence Project
{yokoi,jun.suzuki,inui}@ecei.tohoku.ac.jp

[3] Preferred Networks, Inc. [4] The Institute of Statistical Mathematics
sosk@preferred.jp fukumizu@ism.ac.jp

## Abstract

In this paper, we propose a new kernel-based co-occurrence measure that can be applied to sparse linguistic expressions (e.g., sentences) with a very short learning time, as an alternative to pointwise mutual information (PMI). As well as deriving PMI from mutual information, we derive this new measure from the Hilbert–Schmidt independence criterion (HSIC); thus, we call the new measure the pointwise HSIC (PHSIC). PHSIC can be interpreted as a smoothed variant of PMI that allows various similarity metrics (e.g., sentence embeddings) to be plugged in as kernels. Moreover, PHSIC can be estimated by simple and fast (linear in the size of the data) matrix calculations regardless of whether we use linear or nonlinear kernels. Empirically, in a dialogue response selection task, PHSIC is learned thousands of times faster than an RNN-based PMI while outperforming PMI in accuracy. In addition, we also demonstrate that PHSIC is beneficial as a criterion of a data selection task for machine translation owing to its ability to give high (low) scores to a consistent (inconsistent) pair with other pairs.

## 1 Introduction

Computing the co-occurrence strength between two linguistic expressions is a fundamental task in natural language processing (NLP). For example, in collocation extraction (Manning and Schütze, 1999), word bigrams are collected from corpora and then strongly co-occuring bigrams (e.g., "New York") are found. In dialogue response selection (Lowe et al., 2015), pairs comprising a context and its response sentence are collected from dialogue corpora and the goal is to rank the candidate responses for each given context sentence. In either case, a set of linguistic expression pairs $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{n}$ is first collected and then the co-occurrence strength of a (new) pair $(x, y)$ is computed.

| | | Robustness to Sparsity | Learning Time |
|---|---|---|---|
| **PMI** | | | |
| $\log \frac{n \cdot c(x,y)}{\sum_{y'} c(x,y') \sum_{x'} c(x',y)}$ | Eq. 1 | | ✓ |
| $\log \frac{\widehat{\mathbf{P}}_{\mathrm{RNN}}(y\|x)}{\widehat{\mathbf{P}}_{\mathrm{RNN}}(y)}$ | Eq. 2 | ✓ | |
| **PHSIC** | | | |
| $(\phi(x) - \overline{\phi(x)})^\top \widehat{C}_{XY} (\psi(y) - \overline{\psi(y)})$ | Sec. 5.1 | ✓ | ✓ |
| $(\boldsymbol{a} - \overline{\boldsymbol{a}})^\top \widehat{C}_{\mathrm{ICD}} (\boldsymbol{b} - \overline{\boldsymbol{b}})$ | Sec. 5.2 | ✓ | ✓ |

Table 1: The proposed co-occurrence norm, PHSIC, eliminates the trade-off between robustness to data sparsity and learning time, which PMI has (Section 1).

Pointwise mutual information (PMI) (Church and Hanks, 1989) is frequently used to model the co-occurrence strength of linguistic expression pairs. There are two typical types of PMI estimation (computation) method. One is a counting-based estimator using maximum likelihood estimation, sometimes with smoothing techniques, for example,

$$\widehat{\mathrm{PMI}}_{\mathrm{MLE}}(x, y; \mathcal{D}) = \log \frac{n \cdot c(x,y)}{\sum_{y'} c(x,y') \sum_{x'} c(x',y)}, \tag{1}$$

where $c(x, y)$ denotes the frequency of the pair $(x, y)$ in given data $\mathcal{D}$. This is easy to compute and is commonly used to measure co-occurrence between words, such as in collocation extraction[1]; however, when data $\mathcal{D}$ is sparse, i.e., when $x$ or $y$ is a phrase or sentence, this approach is unrealistic. The second method uses recurrent neural networks (RNNs). Li et al. (2016) proposed to em-

[1] In collocation extraction, simple counting $c(x, y) \propto \widehat{\mathbf{P}}(x, y)$, rather than PMI, ranks undesirable function-word pairs (e.g., "of the") higher (Manning and Schütze, 1999).

*Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1763–1775
Brussels, Belgium, October 31 - November 4, 2018. ©2018 Association for Computational Linguistics

ploy PMI to suppress *dull* responses for utterance generation in dialogue systems[2]. They estimated $\mathbf{P}(y)$ and $\mathbf{P}(y|x)$ using RNN language models and estimated PMI as follows:

$$\widehat{\mathrm{PMI}}_{\mathrm{RNN}}(x, y; \mathcal{D}) = \log \frac{\widehat{\mathbf{P}}_{\mathrm{RNN}}(y|x)}{\widehat{\mathbf{P}}_{\mathrm{RNN}}(y)}. \quad (2)$$

This way of estimating PMI is applicable to sparse language expressions; however, learning RNN language models is computationally costly.

To eliminate this trade-off between robustness to data sparsity and learning time, in this study we propose a new kernel-based co-occurrence measure, which we call the *pointwise Hilbert–Schmidt independence criterion (PHSIC)* (see Table 1). Our contributions are as follows:

- We formalize PHSIC, which is derived from HSIC (Gretton et al., 2005), a kernel-based dependence measure, in the same way that PMI is derived from mutual information (Section 3).
- We give an intuitive explanation why PHSIC is robust to data sparsity. PHSIC is a "smoothed variant of PMI", which allows various similarity metrics to be plugged in as kernels (Section 4).
- We propose fast estimators of PHSIC, which are reduced to a simple and fast matrix calculation regardless of whether we use linear or nonlinear kernels (Section 5).
- We empirically confirmed the effectiveness of PHSIC, i.e., its robustness to data sparsity and learning time, in two different types of experiment, a dialogue response selection task and a data selection task for machine translation (Section 6).

## 2 Problem Setting

Let $X$ and $Y$ denote random variables on $\mathcal{X}$ and $\mathcal{Y}$, respectively. In this paper, we deal with the tasks of taking a set of linguistic expression pairs

$$\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{n} \underset{\text{i.i.d.}}{\sim} \mathbf{P}_{XY}, \quad (3)$$

which is regarded as a set of i.i.d. samples drawn from a joint distribution $\mathbf{P}_{XY}$, and then measuring the "co-occurrence strength" for each given pair $(x, y) \in \mathcal{X} \times \mathcal{Y}$. Such tasks include collocation extraction and dialogue response selection (Section 1).

## 3 Pointwise HSIC

In this section, we give the formal definition of PHSIC, a new kernel-based co-occurrence measure. We show a summary of this section in Table 2. Intuitively, PHSIC is a "kernelized variant of PMI."

### 3.1 Dependence Measure

As a preliminary step, we introduce the simple concept of dependence (see **Dependence Measure** in Table 2). Recall that random variables $X$ and $Y$ are *independent* if and only if the joint probability density $\mathbf{P}_{XY}$ and the product of the marginals $\mathbf{P}_X\mathbf{P}_Y$ are equivalent. Therefore, we can measure the *dependence* between random variables $X$ and $Y$ via the difference between $\mathbf{P}_{XY}$ and $\mathbf{P}_X\mathbf{P}_Y$.

Both the mutual information and the Hilbert–Schmidt independence criterion, to be described below, are such dependence measures.

### 3.2 MI and PMI

We briefly review the well-known mutual information and PMI (see **MI & PMI** in Table 2).

The *mutual information (MI)*[3] between two random variables $X$ and $Y$ is defined by

$$\mathrm{MI}(X, Y) := \mathrm{KL}[\mathbf{P}_{XY} \| \mathbf{P}_X\mathbf{P}_Y] \quad (4)$$

(Cover and Thomas, 2006), where $\mathrm{KL}[\cdot\|\cdot]$ denotes the Kullback–Leibler (KL) divergence. Thus, $\mathrm{MI}(X, Y)$ is the degree of dependence between $X$ and $Y$ measured by the KL divergence between $\mathbf{P}_{XY}$ and $\mathbf{P}_X\mathbf{P}_Y$.

Here, by definition of the KL divergence, MI can be represented in the form of the expectation over $\mathbf{P}_{XY}$, i.e., the summation over all possible pairs $(x, y) \in \mathcal{X} \times \mathcal{Y}$:

$$\mathrm{MI}(X, Y) = \underset{(x,y)}{\mathbf{E}}\left[\log \frac{\mathbf{P}_{XY}(x, y)}{\mathbf{P}_X(x)\mathbf{P}_Y(y)}\right]. \quad (5)$$

The shaded part in Equation (5) is actually the *pointwise mutual information (PMI)* (Church and Hanks, 1989):

$$\mathrm{PMI}(x, y; X, Y) := \log \frac{\mathbf{P}_{XY}(x, y)}{\mathbf{P}_X(x)\mathbf{P}_Y(y)}. \quad (6)$$

Therefore, $\mathrm{PMI}(x, y)$ can be thought of as the contribution of $(x, y)$ to $\mathrm{MI}(X, Y)$.

[2] In dialogue response selection or generation, a simple conditional probability $\widehat{\mathbf{P}}(y|x)$, rather than PMI, ranks *dull* responses (e.g., "I don't know.") higher (Li et al., 2016).

[3] Conventionally, mutual information is denoted by $I(X; Y)$; in this paper, however, for notational consistency, mutual information is denoted by $\mathrm{MI}(X, Y)$.

|  | **Dependence Measure**<br>the dependence between $X$ and $Y$<br>(the difference between $\mathbf{P}_{XY}$ and $\mathbf{P}_X\mathbf{P}_Y$) | **Co-occurrence Measure**<br>the contribution of $(x, y)$<br>to the dependence between $X$ and $Y$ |
|---|---|---|
| **MI & PMI** | $\mathrm{MI}(X, Y) = \mathrm{KL}[\mathbf{P}_{XY} \Vert \mathbf{P}_X\mathbf{P}_Y]$<br>$= \mathop{\mathbf{E}}_{(x,y)}\left[\log \frac{\mathbf{P}_{XY}(x,y)}{\mathbf{P}_X(x)\mathbf{P}_Y(y)}\right]$ | $\mathrm{PMI}(x, y; X, Y)$<br>$= \log \frac{\mathbf{P}_{XY}(x,y)}{\mathbf{P}_X(x)\mathbf{P}_Y(y)}$ |
| **HSIC & PHSIC** | $\mathrm{HSIC}(X, Y; k, \ell) = \mathrm{MMD}^2_{k,\ell}[\mathbf{P}_{XY}, \mathbf{P}_X\mathbf{P}_Y]$<br>$= \mathop{\mathbf{E}}_{(x,y)}\left[(\phi(x) - m_X)^\top C_{XY}(\psi(y) - m_Y)\right]$<br>$= \mathop{\mathbf{E}}_{(x,y)}\left[\mathop{\mathbf{E}}_{(x',y')}[\widetilde{k}(x, x')\widetilde{\ell}(y, y')]\right]$ | $\mathrm{PHSIC}(x, y; X, Y, k, \ell)$<br>$= (\phi(x) - m_X)^\top C_{XY}(\psi(y) - m_Y)$<br>$= \mathop{\mathbf{E}}_{(x',y')}[\widetilde{k}(x, x')\widetilde{\ell}(y, y')]$ |

Table 2: Relationship between the mutual information (MI), the pointwise mutual information (PMI), the Hilbert–Schmidt independence criterion (HSIC), and the pointwise HSIC (PHSIC). As well as defining PMI as the contribution to MI, we define PHSIC as the contribution to HSIC. In short, PHSIC is a "kernelized PMI" (Section 3).

### 3.3 HSIC and PHSIC

As seen in the previous section, PMI can be derived from MI. Here, we consider replacing MI with the Hilbert–Schmidt independence criterion (HSIC). Then, in analogy with the relationship between PMI and MI, we derive PHSIC from HSIC (see **HSIC & PHSIC** in Table 2).

Let $k \colon \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ and $\ell \colon \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ denote positive definite kernels on $\mathcal{X}$ and $\mathcal{Y}$, respectively (intuitively, they are similarity functions between linguistic expressions). The *Hilbert–Schmidt independence criterion (HSIC)* (Gretton et al., 2005), a kernel-based dependence measure, is defined by

$$\mathrm{HSIC}(X, Y; k, \ell) := \mathrm{MMD}^2_{k,\ell}[\mathbf{P}_{XY}, \mathbf{P}_X\mathbf{P}_Y], \quad (7)$$

where $\mathrm{MMD}[\cdot, \cdot]$ denotes the maximum mean discrepancy (MMD) (Gretton et al., 2012), which measures the difference between random variables on a kernel-induced feature space. Thus, $\mathrm{HSIC}(X, Y; k, \ell)$ is the degree of dependence between $X$ and $Y$ measured by the MMD between $\mathbf{P}_{XY}$ and $\mathbf{P}_X\mathbf{P}_Y$, while MI is measured by the KL divergence (Equation (4)).

Analogous to MI in Equation (5), HSIC can be represented in the form of the expectation on $\mathbf{P}_{XY}$ by a simple deformation:

$$\begin{aligned} &\mathrm{HSIC}(X, Y; k, \ell) \\ &= \mathop{\mathbf{E}}_{(x,y)}\left[(\phi(x) - m_X)^\top C_{XY}(\psi(y) - m_Y)\right] && (8) \\ &= \mathop{\mathbf{E}}_{(x,y)}\left[\mathop{\mathbf{E}}_{(x',y')}[\widetilde{k}(x, x')\widetilde{\ell}(y, y')]\right], && (9) \end{aligned}$$

where

$$\phi(x) := k(x, \cdot), \quad \psi(y) := \ell(y, \cdot), \quad (10)$$

$$m_X := \mathbf{E}_x[\phi(x)], \quad m_Y := \mathbf{E}_y[\psi(y)], \quad (11)$$

$$C_{XY} := \mathop{\mathbf{E}}_{(x,y)}\left[(\phi(x) - m_X)(\psi(y) - m_Y)^\top\right], \quad (12)$$

$$\begin{aligned} \widetilde{k}(x, x') &:= k(x, x') - \mathbf{E}_{x'}[k(x, x')] \\ &\quad - \mathbf{E}_x[k(x, x')] + \mathbf{E}_{x,x'}[k(x, x')]. \quad (13) \end{aligned}$$

At first glance, these equations are somewhat complicated; however, the estimators of PHSIC we actually use are reduced to a simple matrix calculation in Section 5. Unlike MI in Equation (5), HSIC has two representations: Equation (8) is the representation in feature space and Equation (9) is the representation in data space.

Similar to the relationship between MI and PMI (Section 3.2), we define the *pointwise Hilbert–Schmidt independence criterion (PHSIC)* by the shaded parts in Equations (8) and (9):

$$\begin{aligned} &\mathrm{PHSIC}(x, y; X, Y, k, \ell) \\ &:= (\phi(x) - m_X)^\top C_{XY}(\psi(y) - m_Y) && (14) \\ &= \mathop{\mathbf{E}}_{(x',y')}[\widetilde{k}(x, x')\widetilde{\ell}(y, y')]. && (15) \end{aligned}$$

Namely, $\mathrm{PHSIC}(x, y)$ is defined as the contribution of $(x, y)$ to $\mathrm{HSIC}(X, Y)$.

In summary, we define PHSIC such that "MI:PMI = HSIC:PHSIC" holds (see Table 2).

## 4 PHSIC as Smoothed PMI

This section gives an intuitive explanation for the first feature of PHSIC, i.e., the robustness to data sparsity, using Table 3. In short, we show that PHSIC is a "smoothed variant of PMI."

First, the maximum likelihood estimator of PMI

|  | add scores | deduct scores |
|---|---|---|
| $\widehat{\mathrm{PMI}}(x,y;\mathcal{D}) = \log \dfrac{n \cdot \sum_i \mathbb{I}[x{=}x_i \wedge y{=}y_i]}{\sum_i \mathbb{I}[x{=}x_i] \sum_i \mathbb{I}[y{=}y_i]}$ | $\begin{array}{c}(x,y)\\ \| \quad \| \\ \mathcal{D}=\{\dots,(x_i,y_i),\dots\}\end{array}$ | $\begin{array}{cc}(x,y) & (x,y)\\ \| \quad \nparallel & \nparallel \quad \| \\ \{\dots,(x_i,y_i),\dots, & (x_i,y_i),\dots\}\end{array}$ |
| $\widehat{\mathrm{PHSIC}}(x,y;\mathcal{D},k,\ell) = \frac{1}{n}\sum_i \hat{\hat{k}}(x,x_i)\hat{\hat{\ell}}(y,y_i)$ | $\begin{array}{cc}(x,y) & (x,y)\\ \wr \quad \wr & \not\wr \quad \not\wr \\ \{\dots,(x_i,y_i),\dots, & (x_i,y_i),\dots\}\end{array}$ | $\begin{array}{cc}(x,y) & (x,y)\\ \wr \quad \not\wr & \not\wr \quad \wr \\ \{\dots,(x_i,y_i),\dots, & (x_i,y_i),\dots\}\end{array}$ |

Table 3: Comparison of estimators of PMI and PHSIC in terms of methods of matching the given $(x, y)$ and the observed $(x_i, y_i)$ in $\mathcal{D}$. PMI matches them in an exact manner, while PHSIC smooths the matching using kernels. Therefore, PHSIC is expected to be robust to data sparsity (Section 4).

in Equation (1) can be rewritten as

$$\widehat{\mathrm{PMI}}(x,y;\mathcal{D}) = \log \frac{n \cdot \sum_i \mathbb{I}[x{=}x_i \wedge y{=}y_i]}{\sum_i \mathbb{I}[x{=}x_i] \sum_i \mathbb{I}[y{=}y_i]}, \quad (16)$$

where $\mathbb{I}[\text{condition}] = 1$ if the condition is true and $\mathbb{I}[\text{condition}] = 0$ otherwise. According to Equation (16), $\widehat{\mathrm{PMI}}(x, y)$ is the amount computed by repeating the following operation (see the first row in Table 3):

> *collate the given $(x, y)$ and the observed $(x_i, y_i)$ in $\mathcal{D}$ in order, and add the scores if $(x, y)$ and $(x_i, y_i)$ match exactly or deduct the scores if either the $x$ side or the $y$ side (but nor both) matches.*

Moreover, an estimator of PHSIC in data space (Equation (15)) is

$$\widehat{\mathrm{PHSIC}}(x,y;\mathcal{D},k,\ell) = \frac{1}{n}\sum_i \hat{\hat{k}}(x,x_i)\hat{\hat{\ell}}(y,y_i), \quad (17)$$

where $\hat{\hat{k}}(\cdot,\cdot)$ and $\hat{\hat{\ell}}(\cdot,\cdot)$ are similarity functions centered on the data[4]. According to Equation (17), $\widehat{\mathrm{PHSIC}}(x, y)$ is the amount computed by repeating the following operation (see the second row in Table 3):

> *collate the given $(x, y)$ and the observed $(x_i, y_i)$ in $\mathcal{D}$ in order, and add the scores if the similarities on the $x$ and $y$ sides are both higher (both $\hat{\hat{k}}(x, x_i) > 0$ and $\hat{\hat{\ell}}(y, y_i) > 0$ hold)[5] or deduct the scores if the similarities on either the $x$ or $y$ sides are similar but those on the other side are not similar.*

As described above, when comparing the estimators of PMI and PHSIC from the viewpoint of "methods of matching the given $(x, y)$ and the observed $(x_i, y_i)$," it is understood that PMI matches them in an exact manner, while PHSIC smooths the matching using kernels (similarity functions).

With this mechanism, even for completely unknown pairs, it is possible to estimate the co-occurrence strength by referring to observed pairs through the kernels. Therefore, PHSIC is expected to be robust to data sparsity and can be applied to phrases and sentences.

**Available Kernels for PHSIC** In NLP, a variety of similarity functions (i.e., positive definite kernels) are available. We can freely utilize such resources, such as cosine similarity between sentence embeddings. For a more detailed discussion, see Appendix A.

## 5 Empirical Estimators of PHSIC

Recall that we have two types of empirical estimator of PMI, the maximum likelihood estimator (Equation (1)) and the RNN-based estimator (Equation (2)). In this section, we describe how to rapidly estimate PHSIC from data. When using the linear kernel or cosine similarity (e.g., cosine similarity between sentence embeddings), PHSIC can be efficiently estimated in feature space (Section 5.1). When using a nonlinear kernel such as the Gaussian kernel, PHSIC can also be estimated efficiently in data space via a simple matrix decomposition (Section 5.2).

### 5.1 Estimation Using Linear Kernel or Cosine

When using the linear kernel or cosine similarity, the estimator of PHSIC in feature space (14) is as follows:

[4] To be exact, $\hat{\hat{k}}(x, x') := k(x, x') - \frac{1}{n}\sum_{j=1}^{n} k(x, x_j) - \frac{1}{n}\sum_{i=1}^{n} k(x_i, x') + \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} k(x_i, x_j)$, which is an estimator of the centered kernel $\tilde{k}(x, x')$ in Equation (13).

[5] In addition, the scores are added if the similarity on the $x$ side and that on the $y$ side are both lower, that is, if $\hat{\hat{k}}(x, x_i) < 0$ and $\hat{\hat{\ell}}(y, y_i) < 0$ hold.

$$\widehat{\mathrm{PHSIC}}_{\mathrm{feature}}(x, y; \mathcal{D}, k, \ell)$$
$$= (\phi(x) - \overline{\phi(x)})^\top \widehat{C}_{XY} (\psi(y) - \overline{\psi(y)}), \quad (18)$$

where

$$\phi(x) = \begin{cases} x & (k(x, x') = x^\top x') \\ x/\|x\| & (k(x, x') = \cos(x, x')) \end{cases}, \quad (19)$$

$$\overline{\phi(x)} := \frac{1}{n}\sum_{i=1}^{n} \phi(x_i), \quad \overline{\psi(y)} := \frac{1}{n}\sum_{i=1}^{n} \psi(y_i), \quad (20)$$

$$\widehat{C}_{XY} := \frac{1}{n}\sum_{i=1}^{n} \phi(x_i)\psi(y_i)^\top - \overline{\phi(x)}\,\overline{\psi(y)}^\top. \quad (21)$$

Generally in kernel methods, a feature map $\phi(\cdot)$ induced by a kernel $k(\cdot, \cdot)$ is unknown or high-dimensional and it is difficult to compute estimated values in feature space[6]. However, when we use the linear kernel or cosine similarity, feature maps can be explicitly determined (Equation (19)).

**Computational Cost** When learning Equation (18) with feature maps $\phi\colon \mathcal{X} \to \mathbb{R}^d$ and $\psi\colon \mathcal{Y} \to \mathbb{R}^d$, computing the vectors $\overline{\phi(x)}, \overline{\psi(y)} \in \mathbb{R}^d$ and the matrix $\widehat{C}_{XY} \in \mathbb{R}^{d \times d}$ takes $\mathcal{O}(nd^2)$ time and $\mathcal{O}(nd)$ space (linear in the size of the input, $n$). When estimating PHSIC$(x, y)$, computing $\phi(x), \psi(y) \in \mathbb{R}^d$ and Equation (18) takes $\mathcal{O}(d^2)$ time (constant; does not depend on the size of the input, $n$).

### 5.2 Estimation Using Nonlinear Kernels

When using a nonlinear kernel such as the Gaussian kernel, it is necessary to estimate PHSIC in data space. Using a simple matrix decomposition, this can be achieved with the same computational cost as the estimation in feature space. See Appendix B for a detailed derivation.

## 6 Experiments

In this section, we provide empirical evidence for the greater effectiveness of PHSIC than PMI, i.e., a very short learning time and robustness to data sparsity. Among the many potential applications of PHSIC, we choose two fundamental scenarios, (re-)ranking/classification and data selection.

- In the ranking/classification scenario (measuring the co-occurrence strength of *new data pairs* with reference to observed pairs), PHSIC is applied as a criterion for the dialogue response selection task (Section 6.2).
- In the data selection/filtering scenario (ordering the entire set of *observed data pairs* according to the co-occurrence strength), PHSIC is also applied as a criterion for data selection in the context of machine translation (Section 6.3).

### 6.1 PHSIC Settings

To take advantage of recent developments in representation learning, we used several pre-trained models for encoding sentences into vectors and several kernels between these vectors for PHSIC.

**Encoders** As sentence encorders, we used two pre-trained models without fine-tuning. First, the sum of the word vectors effectively represents a sentence (Mikolov et al., 2013a):

$$\boldsymbol{x} = \textstyle\sum_{w \in x} \mathrm{vec}(w), \quad \boldsymbol{y} = \textstyle\sum_{w \in y} \mathrm{vec}(w). \quad (22)$$

For $\mathrm{vec}(\cdot)$, we used the pre-trained **fastText** model[7], which is a high-accuracy and popular word embedding model (Bojanowski et al., 2017); models in 157 languages are publicly distributed (Grave et al., 2018). Second, we also used a DNN-based sentence encoder, called the universal sentence encoder (Cer et al., 2018), which utilizes the deep averaging network (**DAN**) (Iyyer et al., 2015). The pre-trained model for English sentences we used is publicly available[8].

**Kernels** As kernels between these vectors, we used cosine similarity (**cos**)

$$k(\boldsymbol{x}, \boldsymbol{x}') = \cos(\boldsymbol{x}, \boldsymbol{x}') \quad (23)$$

and the Gaussian kernel (also known as the radial basis function kernel; **RBF** kernel)

$$k(\boldsymbol{x}, \boldsymbol{x}') = \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{x}'\|_2^2}{2\sigma^2}\right), \quad (24)$$

and similarly for $\ell(\boldsymbol{y}, \boldsymbol{y}')$. The experiments are ran with hyperparameter $\sigma = 1.0$ for the RBF kernel, and $d = 100$ for incomplete Cholesky decomposition (for more detail, see Section B).

### 6.2 Ranking: Dialogue Response Selection

In the first experiment, we applied PHSIC as a ranking criterion of the task of dialogue response

[6] One of the characteristics of kernel methods is that an intractable estimation in feature space is replaced with an efficient estimation in data space.

[7] https://fasttext.cc/docs/en/english-vectors.html, https://fasttext.cc/docs/en/crawl-vectors.html

[8] https://www.tensorflow.org/hub/modules/google/universal-sentence-encoder/1

selection (Lowe et al., 2015); in the task, pairs comprising a context (previous utterance sequence) and its response are collected from dialogue corpora and the goal is to rank the candidate responses for each given context sentence.

The task entails sentence sequences (very sparse linguistic expressions); moreover, Li et al. (2016) pointed out that (RNN-based) PMI has a positive impact on suppressing *dull* responses (e.g., "I don't know.") in dialogue systems. Therefore, PHSIC, another co-occurrence measure, is also expected to be effective for this. With this setting, where the validity of PMI is confirmed, we investigate whether PHSIC can replace RNN-based PMI in terms of both learning time and robustness to data sparsity.

### Experimental Settings

**Dataset** For the training data, we gathered approximately $5 \times 10^5$ reply chains from Twitter, following Sordoni et al. (2015)[9]. In addition, we randomly selected $\{10^3, 10^4, 10^5\}$ reply chains from that dataset. Using these small subsets, we confirmed the effect of the difference in the size of the training set (data sparseness) on the learning time and predictive performance.

For validation and test data, we used a small (approximately 2000 pairs each) but highly reliable dataset created by Sordoni et al. (2015)[10], which consists only of conversations given high scores by human annotators. Therefore, this set was not expected to include *dull* responses.

For each dataset, we converted each context-message-response triple into a context-response pair by concatenating the context and message following Li et al. (2016). In addition, to convert the test set (positive examples) to ten-choice multiple-choice questions, we shuffled the combinations of context and response to generate pseudo-negative examples.

**Evaluation Metrics** We adopted the following evaluation metrics for the task: (i) `ROC-AUC` (the area under the receiver operating characteristic curve), (ii) `MRR` (the mean reciprocal rank), and (iii) `Recall@{1,2}`.

| | *Config* | | | **Size of Training Set** $n$ | | | |
|---|---|---|---|---|---|---|---|
| | | | | $10^3$ | $10^4$ | $10^5$ | $5\times10^5$ |
| **RNN-PMI** | *Dim.* | *Init.* | | | | | |
| | | | Total | 20.6 | 99.2 | 634.3 | 4042.5 |
| | 300 | `fastText` | $\widehat{\mathbf{P}}(y)$ | 8.0 | 23.6 | 294.6 | 1710.1 |
| | | | $\widehat{\mathbf{P}}(y\|x)$ | 12.6 | 75.6 | 339.7 | 2332.4 |
| | | | Total | 49.0 | 162.0 | 1751.3 | 13054.9 |
| | 1200 | `fastText` | $\widehat{\mathbf{P}}(y)$ | 16.3 | 57.2 | 671.0 | 5512.1 |
| | | | $\widehat{\mathbf{P}}(y\|x)$ | 32.7 | 104.8 | 1080.3 | 7542.8 |
| **PHSIC** | *Encoder* | *Kernel* | | | | | |
| | `fastText` | `cos` | | **0.0** | **0.1** | **0.5** | **2.8** |
| | `DAN` | `cos` | | **0.0** | **0.1** | **0.4** | **1.8** |

Table 4: **Learning time** [s] for each model and each size of training set for the dialogue response task. Each row denotes a model; each column denotes the number of training data $n$. The text appended to each baseline model denotes the number of dimension of hidden layers (*Dim.*) and the method of initialization the embedding layer (*Init.*). The text appended to each proposed model denotes the pre-trained models used to encode sentences into vectors (*Encoder*) and the kernel between these vectors (*Kernel*). The best result (the shortest learning time) in each column is in bold.

**Experimental Procedure** We used the following procedure: (i) train the model with a set of context-response pairs $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{n}$; (ii) for each context sentence $x$ in the test data, rank the candidate responses $\{y_j\}_{j=1}^{10}$ by the model; and (iii) report three evaluation metrics.

**Baseline Measures** As baseline measures, both (1) an RNN language model $\widehat{\mathbf{P}}_{\mathrm{RNN}}(y)$ (Mikolov et al., 2010) and (2) a conditional RNN language model $\widehat{\mathbf{P}}_{\mathrm{RNN}}(y|x)$ (Sutskever et al., 2014) were trained, and (3) PMI based on these language models, **RNN-PMI**, was also used for experiments (see Equation (2)). We trained these models with all combinations of the following settings: (a) the number of dimensions of the hidden layers being 300 or 1200 and (b) the initialization of the embedding layer being **`random`** (uniform on $[-0.1, 0.1]$) or **`fastText`**. For more detailed settings, see Appendix C.

### Experimental Results

**Learning Time** Table 4 shows the experimental results of the learning time[11]. Regardless of the size of the training set $n$, the learning time for

[9] We collected tweets after 2017 for our training set to avoid duplication with the test set, which contains tweets from the year 2012.

[10] `https://www.microsoft.com/en-us/download/details.aspx?id=52375`

[11] The computing environment was as follows:
(i) CPU: Xeon E5-1650-v3 (3.5 GHz, 6 Cores);
(ii) GPU: GTX 1080 (8 GB).

PHSIC is much shorter than that of the RNN-based method. For example, even when the size of the training set $n$ is $5 \times 10^5$, PHSIC is approximately 1400–4000 times faster than RNN-based PMI. This is because the estimators of PHSIC are reduced to a deterministic and efficient matrix calculation (Section 5), whereas neural network-based models involve the sequential optimization of parameters via gradient descent methods.

**Robustness to Data Sparsity** Table 5 shows the experimental results of the predictive performance. When the size of the training data is small ($n = 10^3, 10^4$), that is, when the data is extremely sparse, the predictive performance of PHSIC hardly deteriorates while that of PMI rapidly decays as the number of data decreases. This indicates that PHSIC is more robust to data sparsity than RNN-based PMI owing to the effect of kernels. Moreover, PHSIC with the simple cosine kernel outperforms the RNN-based model regardless of the number of data, while the learning time of PHSIC is thousands of times shorter than those of the baseline methods (Section 6.2).

Additionally we report Spearman's rank correlation coefficient between models to verify whether PHSIC shows similar behavior to PMI. See Appendix D for more detail.

### 6.3 Data Selection for Machine Translation

The aim of our second experiment was to demonstrate that PHSIC is also beneficial as a criterion of data selection. To achieve this, we attempted to apply PHSIC to a parallel corpus filtering task that has been intensively discussed in recent (neural) machine translation (MT, NMT) studies. This task was first adopted as a shared task in the third conference on machine translation (WMT 2018)[12].

Several existing parallel corpora, especially those automatically gathered from large-scale text data, such as the Web, contain unacceptable amounts of noisy (low-quality) sentence pairs that greatly affect the translation quality. Therefore, the development of an effective method for parallel corpus filtering would potentially have a large influence on the MT community; discarding such noisy pairs may improve the translation quality and shorten the training time.

We expect PHSIC to give low scores to *exceptional* sentence pairs (misalignments or missing translations) during the selection process because PHSIC assigns low scores to pairs that are highly inconsistent with other pairs (see Section 4). Note that applying RNN-based PMI to a parallel corpus selection task is unprofitable since obtaining RNN-based PMI also has an identical computational cost for training a sequence-to-sequence model for MT, and thus, we cannot expect a reduction of the total training time.

#### Experimental Settings

**Dataset** We used the ASPEC-JE corpus[13], which is an official dataset used for the MT-evaluation shared task held in the fourth workshop on Asian translation (WAT 2017)[14] (Nakazawa et al., 2017). ASPEC-JE consists of approximately three million (3M) Japanese–English parallel sentences from scientific paper abstracts. As discussed by Kocmi et al. (2017), ASPEC-JE contains many low-quality parallel sentences that have the potential to significantly degrade the MT quality. In fact, they empirically revealed that using only the reliable part of the training parallel corpus significantly improved the translation quality. Therefore, ASPEC-JE is a suitable dataset for evaluating the data selection ability.

**Model** For our data selection evaluation, we selected the Transformer architecture (Vaswani et al., 2017) as our baseline NMT model, which is widely-used in the NMT community and known as one of the current state-of-the-art architectures. We utilized `fairseq`[15], a publicly available tool for neural sequence-to-sequence models, for building our models.

**Experimental Procedure** We used the following procedure for this evaluation: (1) rank all parallel sentences in a given parallel corpus according to each criterion, (2) extract the top $K$ ranked parallel sentences, (3) train the NMT model using the extracted parallel sentences, and (4) evaluate the translation quality of the test data using a typical MT automatic evaluation measure, i.e., BLEU (Papineni et al., 2002)[16]. In our experiments we evaluated PHSIC with $K$ = 0.5M and 1M.

[12] http://www.statmt.org/wmt18/parallel-corpus-filtering.html

[13] http://lotus.kuee.kyoto-u.ac.jp/ASPEC/

[14] http://lotus.kuee.kyoto-u.ac.jp/WAT/WAT2017/

[15] https://github.com/pytorch/fairseq

[16] We used multi-bleu.perl in the Moses tool (https://github.com/moses-smt/mosesdecoder).

| Models | *Config* |  | Size of Training Set $n$ |  |  |  |
|---|---|---|---|---|---|---|
|  |  |  | $10^3$ | $10^4$ | $10^5$ | $5 \times 10^5$ |
| Chance Level |  |  | .50; .29; .10, .20 | .50; .29; .10, .20 | .50; .29; .10, .20 | .50; .29; .10, .20 |
|  | *Dim.* | *Init.* |  |  |  |  |
| $\widehat{\mathbf{P}}_{\mathrm{RNN}}(y)$ | 1200 | `fastText` | .50; .29; .10, .21 | .50; .30; .11, .21 | .50; .30; .10, .21 | .50; .30; .13, .25 |
| $\widehat{\mathbf{P}}_{\mathrm{RNN}}(y\|x)$ | 1200 | `fastText` | .50; .29; .10, .21 | .50; .30; .10, .21 | .52; .31; .11, .23 | .54; .32; .13, .25 |
| **RNN-PMI** | 300 | `random` | .51; .30; .10, .21 | .51; .30; .11, .22 | .58; .35; .14, .29 | .69; .46; .25, .42 |
|  |  | `fastText` | .51; .29; .09, .20 | .56; .34; .15, .25 | .66; .41; .20, .36 | .76; .56; .36, .54 |
|  | 1200 | `random` | .50; .29; .11, .20 | .51; .30; .10, .19 | .57; .35; .14, .29 | .70; .47; .26, .44 |
|  |  | `fastText` | .51; .30; .11, .20 | .52; .32; .13, .23 | .65; .42; .21, .36 | .75; .54; .34, .52 |
| **PHSIC** | *Encoder* | *Kernel* |  |  |  |  |
|  | `fastText` | `cos` | .61; .38; .17, .33 | .62; .40; .19, .34 | .62; .40; .19, .34 | .62; .40; .19, .34 |
|  | `DAN` | `cos` | **.77**; **.58**; **.40**, **.56** | **.78**; **.57**; **.39**, **.56** | **.78**; **.58**; **.41**, **.57** | **.78**; **.58**; **.40**, **.57** |

Table 5: **Predictive performance** for each model and each training set size for the dialogue response selection task: `ROC-AUC`; `MRR`; `Recall@1,2`. The best result in each column is in bold. The other notation is the same as in Table 4.

| Selection Criteria |  |  | # of Selected Data $K$ |  |  |
|---|---|---|---|---|---|
|  |  |  | 0.5M | 1M | 3M |
| (all the training set) |  |  | - | - | 41.02 |
| Random |  |  | 34.26 | 39.82 | - |
| **fast_align** |  |  | 38.63 | 40.56 | - |
|  | *Encoder* | *Kernel* |  |  |  |
| **PHSIC** | `fastText` | `RBF` | **38.95** | **40.95** | - |

Table 6: **BLEU scores** with the Transformer for each data selection criterion and each size of selected data $K$ for the parallel corpus filtering task."Random" represents the baseline method of selecting sentences at random.

**Baseline Measure** As a baseline measure, we utilize a publicly available script[17] of **fast_align** (Dyer et al., 2013), which is one of the state-of-the-art word aligner. We firstly used the fast_align for the training set $\mathcal{D} = \{(x_i, y_i)\}_i$ to obtain the word alignment between each sentence pair $(x_i, y_i)$, i.e., a set of aligned word pairs with its probabilities. We then computed the co-occurrence score of $(x_i, y_i)$ with sentence-length normalization, i.e., the average log probability of aligned word pairs.

### Experimental Results

Table 6 shows the results of our data selection evaluation. It is common knowledge in NMT that more data gives better performance in general. However, we observed that PHSIC successfully extracted beneficial parallel sentences from the noisy parallel corpus; the result using 1M data extracted from the 3M corpus by PHSIC was almost the same as that using 3M data (the decrease in the BLEU score was only 0.07), whereas that by random extraction reduced the BLEU score by 1.20.

This was actually a surprising result because PHSIC utilizes only monolingual similarity measures (kernels) without any other language resources. This indicates that PHSIC can be applied to a language pair poor in parallel resources. In addition, the surface form and grammatical characteristics between English and Japanese are extremely different[18]; therefore, we expect that PHSIC will work well regardless of the similarity of the language pair.

## 7 Related Work

**Dependence Measures** Measuring independence or dependence (correlation) between two random variables, i.e., estimating dependence from a set of paired data, is a fundamental task in statistics and a very wide area of data science. To measure the complex nonlinear dependence that real data has, we have several choices.

First, information-theoretic MI (Cover and Thomas, 2006) and its variants (Suzuki et al., 2009; Reshef et al., 2011) are the most commonly used dependence measures. However, to the best of our knowledge, there is no practical method of computing MIs for large-multi class high-dimensional

[17] https://github.com/clab/fast_align

[18] For example, word order; English is an SVO (subject-verb-object) language and Japanese is an SOV (subject-object-verb) language.

(having a complex generative model) discrete data, such as sparse linguistic data.

Second, several kernel-based dependence measures have been proposed for measuring nonlinear dependence (Akaho, 2001; Bach and Jordan, 2002; Gretton et al., 2005). The reason why kernel-based dependence measures work well for real data is that they do not explicitly estimate densities, which is difficult for high-dimensional data. Among them, HSIC (Gretton et al., 2005) is popular because it has a simple estimation method, which is used for various tasks such as feature selection (Song et al., 2012), dimensionality reduction (Fukumizu et al., 2009), and unsupervised object matching (Quadrianto et al., 2009; Jagarlamudi et al., 2010). We follow this line.

**Co-occurrence Measures** First, In NLP, PMI (Church and Hanks, 1989) and its variants (Bouma, 2009) are the de facto co-occurrence measures between *dense* linguistic expressions, such as words (Bouma, 2009) and simple narrative-event expressions (Chambers and Jurafsky, 2008). In recent years, positive PMI (PPMI) has played an important role as a component of word vectors (Levy and Goldberg, 2014).

Second, there are several studies in which the pairwise ranking problem has been solved by using deep neural networks (DNNs) in NLP. Li et al. (2016) proposed a PMI estimation using RNN language models; this was used as a baseline model in our experiments (see Section 6.2). Several studies have used DNN-based binary classifiers modeling $\mathbf{P}(C = \text{positive} \mid (x, y))$ to solve the given ranking problem directly (Hu et al., 2014; Yin et al., 2016; Mueller and Thyagarajan, 2016) (these networks are sometimes called Siamese neural networks). Our study focuses on comparing co-occurrence measures. It is unknown whether Siamese NNs capture the co-occurrence strength; therefore we did not deal with Siamese NNs in this paper.

Finally, to the best of our knowledge, Yokoi et al. (2017)'s paper is the first study that suggested converting HSIC to a pointwise measure. The present study was inspired by their suggestion; here, we have (i) provided a formal definition (population) of PHSIC; (ii) analyzed the relationship between PHSIC and PMI; (iii) proposed linear-time estimation methods; and (iv) experimentally verified the computation speed and robustness to data sparsity of PHSIC for practical applications.

## 8 Conclusion

The NLP community has commonly employed PMI to estimate the co-occurrence strength between linguistic expressions; however, existing PMI estimators have a high computational cost when applied to sparse linguistic expressions (Section 1). We proposed a new kernel-based co-occurrence measure, the pointwise Hilbert–Schmidt independent criterion (PHSIC). As well as defining PMI as the contribution to mutual information, PHSIC is defined as the contribution to HSIC; PHSIC is intuitively a "kernelized variant of PMI" (Section 3). PHSIC can be applied to sparse linguistic expressions owing to the mechanism of smoothing by kernels. Comparing the estimators of PMI and PHSIC, PHSIC can be interpreted as a smoothed variant of PMI, which allows various similarity metrics to be plugged in as kernels (Section 4). In addition, PHSIC can be estimated in linear time owing to the efficient matrix calculation, regardless of whether we use linear or nonlinear kernels (Section 5). We conducted a ranking task for dialogue systems and a data selection task for machine translation (Section 6). The experimental results show that (i) the learning of PHSIC was completed thousands of times faster than that of the RNN-based PMI while outperforming it in ranking accuracy (Section 6.2); and (ii) even when using a nonlinear kernel, PHSIC can be applied to a large dataset. Moreover, PHSIC reduces the amount of training data to one third without sacrificing the output translation quality (Section 6.3).

**Future Work** Using the PHSIC estimator in feature space (Equation (18)), we can generate the most appropriate $\psi(y)$ for a given $\phi(x)$ (uniquely, up to scale). That is, if a DNN-based sentence decoder is used, $y$ (a sentence) can be restored from $\psi(y)$ (a feature vector) so that generative models of strong co-occurring sentences can be realized.

## Acknowledgments

We are grateful to anonymous reviewers for their helpful comments. We also thank Weihua Hu for useful discussions, Kenshi Yamaguchi for collecting data, and Paul Reisert for proofreading. This work was supported in part by JSPS KAKENHI Grant Number JP15H01702 and JST CREST Grant Number JPMJCR1513, Japan.

## References

Shotaro Akaho. 2001. A kernel method for canonical correlation analysis. In *IMPS*, pages 1–7.

Francis R. Bach and Michael I. Jordan. 2002. Kernel Independent Component Analysis. *JMLR*, 3(Jul):1–48.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching Word Vectors with Subword Information. *TACL*, 5:135–146.

Gerlof Bouma. 2009. Normalized (Pointwise) Mutual Information in Collocation Extraction. In *GSCL*, pages 31–40.

Razvan C Bunescu and Raymond J Mooney. 2006. Subsequence Kernels for Relation Extraction. In *NIPS*, pages 171–178.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2018. Universal Sentence Encoder. *CoRR*, abs/1803.1.

Nathanael Chambers and Dan Jurafsky. 2008. Unsupervised Learning of Narrative Event Chains. In *ACL*, pages 789–797.

Kenneth Ward Church and Patrick Hanks. 1989. Word Association Norms, Mutual Information, and Lexicography. In *ACL*, pages 76–83.

Michael Collins and Nigel Duffy. 2002. Convolution Kernels for Natural Language. In *NIPS*, pages 625–632.

Thomas M. Cover and Joy A. Thomas. 2006. *Elements of Information Theory*. John Wiley & Sons.

Andrew M Dai and Quoc V Le. 2015. Semi-supervised Sequence Learning. In *NIPS*, pages 3079–3087.

Chris Dyer, Victor Chahuneau, and Noah A Smith. 2013. A Simple, Fast, and Effective Reparameterization of IBM Model 2. In *NAACL-HLT*, pages 644–648.

Shai Fine and Katya Scheinberg. 2001. Efficient SVM Training Using Low-Rank Kernel Representations. *JMLR*, 2(Dec):243–264.

Kenji Fukumizu, Francis R. Bach, and Michael I. Jordan. 2009. Kernel dimension reduction in regression. *Annals of Statistics*, 37(4):1871–1905.

Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomas Mikolov. 2018. Learning Word Vectors for 157 Languages. In *LREC*, pages 3483–3487.

Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. 2012. A Kernel Two-Sample Test. *JMLR*, 13(Mar):723–773.

Arthur Gretton, Olivier Bousquet, Alex Smola, and Bernhard Schölkopf. 2005. Measuring Statistical Dependence with Hilbert-Schmidt Norms. In *ALT*, pages 63–77.

Felix Hill, Kyunghyun Cho, and Anna Korhonen. 2016. Learning Distributed Representations of Sentences from Unlabelled Data. In *NAACL-HLT*, pages 1367–1377.

Sepp Hochreiter and Jrgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780.

Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. 2014. Convolutional Neural Network Architectures for Matching Natural Language Sentences. In *NIPS*, pages 2042–2050.

Mohit Iyyer, Varun Manjunatha, Jordan Boyd-Graber, and Hal Daumé III. 2015. Deep Unordered Composition Rivals Syntactic Methods for Text Classification. In *ACL/IJCNLP*, pages 1681–1691.

Jagadeesh Jagarlamudi, Seth Juarez, and Hal Daumé III. 2010. Kernelized Sorting for Natural Language Processing. In *AAAI*, pages 1020–1025.

Ryan Kiros, Yukun Zhu, Ruslan R. Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-Thought Vectors. In *NIPS*, pages 3294–3302.

Tom Kocmi, Dušan Variš, and Ondřej Bojar. 2017. Cuni nmt system for wat 2017 translation tasks. In *WAT*, pages 154–159.

Omer Levy and Yoav Goldberg. 2014. Neural Word Embedding as Implicit Matrix Factorization. In *NIPS*, pages 2177–2185.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A Diversity-Promoting Objective Function for Neural Conversation Models. In *NAACL-HLT*, pages 110–119.

Ryan Lowe, Nissan Pow, Iulian Serban, and Joelle Pineau. 2015. The Ubuntu Dialogue Corpus: A Large Dataset for Research in Unstructured Multi-Turn Dialogue Systems. In *SIGDIAL*, pages 285–294.

Christopher D. Manning and Hinrich Schütze. 1999. *Foundations of Statistical Natural Language Processing*. MIT press.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Distributed Representations of Words and Phrases and their Compositionality. In *NIPS*, pages 3111–3119.

Tomas Mikolov, Greg Corrado, Kai Chen, and Jeffrey Dean. 2013b. Efficient Estimation of Word Representations in Vector Space. In *Proc. of the Workshop on ICLR*, pages 1–12.

Tomas Mikolov, Martin Karafiat, Lukas Burget, Jan Cernocky, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Interspeech*, pages 1045–1048.

Jeff Mitchell and Mirella Lapata. 2010. Composition in Distributional Models of Semantics. *Cognitive Science*, 34(8):1388–1429.

Alessandro Moschitti. 2006. Making Tree Kernels practical for Natural Language Learning. In *EACL*, volume 6, pages 113–120.

Jonas Mueller and Aditya Thyagarajan. 2016. Siamese Recurrent Architectures for Learning Sentence Similarity. In *AAAI*, 2012, pages 2786–2792.

Toshiaki Nakazawa, Shohei Higashiyama, Chenchen Ding, Hideya Mino, Isao Goto, Hideto Kazawa, Yusuke Oda, Graham Neubig, and Sadao Kurohashi. 2017. Overview of the 4th workshop on asian translation. In *WAT*, pages 1–54.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *ACL*, pages 311–318.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. GloVe: Global Vectors for Word Representation. In *EMNLP*, pages 1532–1543.

Novi Quadrianto, Le Song, and Alex J. Smola. 2009. Kernelized sorting. In *NIPS*, pages 1289–1296.

David N. Reshef, Yakir A. Reshef, Hilary K. Finucane, Sharon R. Grossman, Gilean McVean, Peter J. Turnbaugh, Eric S. Lander, Michael Mitzenmacher, and Pardis C. Sabeti. 2011. Detecting Novel Associations in Large Data Sets. *Science*, 334(6062):1518–1524.

J. Shawe-Taylor and N. Cristianini. 2004. *Kernel Methods for Pattern Analysis*. Cambridge University Press.

Le Song, Alex Smola, Arthur Gretton, Justin Bedo, and Karsten Borgwardt. 2012. Feature Selection via Dependence Maximization. *JMLR*, 13:1393–1434.

Alessandro Sordoni, Michel Galley, Michael Auli, Chris Brockett, Yangfeng Ji, Margaret Mitchell, Jian-Yun Nie, Jianfeng Gao, and Bill Dolan. 2015. A Neural Network Approach to Context-Sensitive Generation of Conversational Responses. In *NAACL-HLT*, pages 196–205.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to Sequence Learning with Neural Networks. In *NIPS*, pages 3104–3112.

Taiji Suzuki, Masashi Sugiyama, Takafumi Kanamori, and Jun Sese. 2009. Mutual information estimation reveals global associations between stimuli and biological processes. *BMC Bioinformatics*, 10(Suppl 1):S52.

Seiya Tokui, Kenta Oono, Shohei Hido, and Justin Clayton. 2015. Chainer: a Next-Generation Open Source Framework for Deep Learning. In *LearningSys*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, ukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *NIPS*, pages 5998–6008.

John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2015. Towards Universal Paraphrastic Sentence Embeddings. In *ICLR*, pages 1–19.

Wenpeng Yin, Hinrich Schütze, Bing Xiang, and Bowen Zhou. 2016. ABCNN: Attention-Based Convolutional Neural Network for Modeling Sentence Pairs. *TACL*, 4(1):259–272.

Sho Yokoi, Daichi Mochihashi, Ryo Takahashi, Naoaki Okazaki, and Kentaro Inui. 2017. Learning Co-Substructures by Kernel Dependence Maximization. In *IJCAI*, pages 3329–3335.

## A Available Kernels for PHSIC

**Similarity between Sentence Vectors** A variety of vector representations of phrases and sentences based on the distributional hypothesis have recently been proposed, including sentence encoders (Kiros et al., 2015; Dai and Le, 2015; Iyyer et al., 2015; Hill et al., 2016; Cer et al., 2018) and the sum of word embeddings; it is known as *additive compositionality* (Mitchell and Lapata, 2010; Mikolov et al., 2013a; Wieting et al., 2015) that we can express the meaning of phrases and sentences well with the sum of word vectors (e.g., `word2vec` (Mikolov et al., 2013b), `GloVe` (Pennington et al., 2014), and `fastText` (Bojanowski et al., 2017)). Note that various pre-trained models of sentence encoders and word embeddings have also been made available.

The cosine of these vectors, which is a positive definite kernel, can be used as a convenient and highly accurate similarity function between phrases or sentences. Other major kernels can also be used, such as the RBF kernel, the Laplacian kernel, and polynomial kernels.

**Structured Kernels** Various structured kernels for NLP, such as tree kernels, which capture fine structure of sentences such as syntax, were devised in the support vector machine era (Collins and Duffy, 2002; Bunescu and Mooney, 2006; Moschitti, 2006).

**Combinations** We can freely combine the previously mentioned kernels because the sum and the product of positive definite kernels are also positive definite kernels (Shawe-Taylor and Cristianini, 2004, Proposition 3.22).

## B Derivation of Fast PHSIC Estimation in Data Space

Although estimators of HSIC and PHSIC depend on kernels $k, \ell$ and data $\mathcal{D}$, hereinafter, we use the following notation for the sake of simplicity:

$$\widehat{\mathrm{HSIC}}(X,Y) := \widehat{\mathrm{HSIC}}(X,Y;\mathcal{D},k,\ell), \qquad (25)$$

$$\widehat{\mathrm{PHSIC}}(x,y) := \widehat{\mathrm{PHSIC}}(x,y;\mathcal{D},k,\ell). \qquad (26)$$

**Naïve Estimation** Fist, an estimator of PHSIC in the data space (15) is

$$\widehat{\mathrm{PHSIC}}_{\mathrm{kernel}}(x,y) = (\boldsymbol{k}-\overline{\boldsymbol{k}})^\top(\tfrac{1}{n}H)(\boldsymbol{\ell}-\overline{\boldsymbol{\ell}}), \quad (27)$$

where $\boldsymbol{k} := (k(x,x_1),\dots,k(x,x_n))^\top \in \mathbb{R}^n$, so as $\boldsymbol{\ell}$; and vector $\overline{\boldsymbol{k}} := \frac{1}{n}K\mathbf{1}$ denotes empirical mean of $\{\boldsymbol{k}_i\}_{i=1}^n$, so as $\overline{\boldsymbol{\ell}}$. This estimation has a large computational cost. When learning, computing the vectors $\overline{\boldsymbol{k}}, \overline{\boldsymbol{\ell}}$ takes $\mathcal{O}(n^2)$ time and $\mathcal{O}(n)$ space. When estimating PHSIC, computing $\boldsymbol{k}, \boldsymbol{\ell}$ and multiplying the matrix $\frac{1}{n}H$ takes $\mathcal{O}(n)$ time.

**Fast Estimation via Incomplete Cholesky Decomposition** Equation (27) has a large computational cost because it is necessary to construct the Gram matrices $K$ and $L \in \mathbb{R}^{n\times n}$. In kernel methods, several methods have been proposed for approximating Gram matrices at low cost without constructing them explicitly, such as *incomplete Cholesky decomposition* (Fine and Scheinberg, 2001).

By incomplete Cholesky decomposition, from data points $\{x_1,\dots,x_n\} \subseteq \mathcal{X}$ and a positive definite kernel $k\colon \mathcal{X}\times\mathcal{X} \to \mathbb{R}$, a matrix $A = (\boldsymbol{a}_1,\dots,\boldsymbol{a}_n)^\top \in \mathbb{R}^{n\times d}$ $(d \ll n)$ can be obtained with $\mathcal{O}(nd^2)$ time complexity. This makes it possible to approximate the Gram matrix $K$ by vectors $\boldsymbol{a}_i \in \mathbb{R}^d$ without configuring the entire of $K$:

$$\boldsymbol{a}_i^\top \boldsymbol{a}_j \approx k(x_i,x_j) \qquad (28)$$

$$AA^\top \approx K. \qquad (29)$$

Also, for HSIC, an efficient approximation method utilizing incomplete Cholesky decomposition has been proposed (Gretton et al., 2005, Lemma 2):

$$\widehat{\mathrm{HSIC}}_{\mathrm{ICD}}(X,Y) = \frac{1}{n^2}\|(HA)^\top B\|_{\mathrm{F}}^2, \qquad (30)$$

where $A = (\boldsymbol{a}_1,\dots,\boldsymbol{a}_n)^\top \in \mathbb{R}^{n\times d}$ is a matrix satisfying $AA^\top \approx K$ computed via incomplete Cholesky decomposition, so as $B$ $(BB^\top \approx L)$. Equation (30) can be represented in the form of the expectation on data points:

$$\widehat{\mathrm{HSIC}}_{\mathrm{ICD}}(X,Y) = \frac{1}{n}\sum_{i=1}^{n}\Big[(\boldsymbol{a}_i-\overline{\boldsymbol{a}})^\top \widehat{C}_{\mathrm{ICD}}(\boldsymbol{b}_i-\overline{\boldsymbol{b}})\Big] \qquad (31)$$

$$\widehat{C}_{\mathrm{ICD}} := \frac{1}{n}(HA)^\top B \in \mathbb{R}^{d\times d}, \qquad (32)$$

where vector $\overline{\boldsymbol{a}} := \frac{1}{n}A^\top\mathbf{1} \in \mathbb{R}^d$ denotes empirical mean of $\{\boldsymbol{a}_i\}_{i=1}^n$, so as $\overline{\boldsymbol{b}} := \frac{1}{n}B^\top\mathbf{1}$.

Recall that $\mathrm{PHSIC}(x,y)$ is the contribution of $(x,y)$ to $\mathrm{HSIC}(X,Y)$ (see Section 3.3); PHSIC then can be efficiently estimated by the shaded part of Equation (31):

$$\widehat{\mathrm{PHSIC}}_{\mathrm{ICD}}(x,y) = (\boldsymbol{a}-\overline{\boldsymbol{a}})^\top \widehat{C}_{\mathrm{ICD}}(\boldsymbol{b}-\overline{\boldsymbol{b}}). \qquad (33)$$

Here, the vector $\boldsymbol{a} \in \mathbb{R}^d$ corresponding to the new $x$ can be calculated by “performing from halfway”

| **Models** | *Config* | | | (A) | (B) | (C) | (D) |
|---|---|---|---|---|---|---|---|
| **RNN-PMI** | *Dim.* | *Init.* | | | | | |
| | 300 | fastText | (A) | – | .42 | .12 | .27 |
| | 1200 | fastText | (B) | .42 | – | .12 | .26 |
| **PHSIC** | *Encoder* | *Kernel* | | | | | |
| | fastText | cos | (C) | .12 | .12 | – | .16 |
| | DAN | cos | (D) | .27 | .26 | .16 | – |

Table 7: **Spearman's** $\rho$ between the co-occurrence scores computed by the models in the dialogue response selection task (Section 6.2). The size of training set $n$ is $5 \times 10^5$. The other notation is the same as in Table 4.

on the incomplete Cholesky decomposition algorithm. Let $x^{(1)}, \ldots, x^{(d)}$ denote the dominant $x_i$s adopted during decomposition algorithm. The $j$th element of $\boldsymbol{a}$ can be computed as follows:

$$\boldsymbol{a}[j] = \left[k(x, x^{(j)}) - \sum_{m=1}^{j-1} \boldsymbol{a}[m] A_{jm}\right] / \, A_{jj}, \quad (34)$$

so as $\boldsymbol{b} \in \mathbb{R}^d$ corresponding to the new $y$. The estimation via incomplete Cholesky decomposition (33) is extremely efficient compared to the naive estimation (27); Equation (33)'s computational complexity is equivalent to the estimation in the feature space (18).

## C Detailed Settings for Learning RNNs

Detailed settings for learning RNNs used in this research are as follows:

- Hidden layers: single layer LSTMs (Hochreiter and Schmidhuber, 1997)
- Vocabulary: words with a frequency: 10 or more ($n = 5 \times 10^5$), 2 or more (otherwise)
- Dropout rate: 0.1 (300-dim), 0.3 (1200-dim)
- Batch size: 64
- Max epoch number: 5 ($n = 5 \times 10^5$), 30 (otherwise)
- Deep learning framework: `Chainer` (Tokui et al., 2015)

## D Correlation Between Models in Dialogue Response Selection Task

Table D shows Spearman's rank correlation coefficient (Spearman's $\rho$) between the co-occurrence scores on the test set computed by the models in the dialogue response selection task (Section 6.2). This shows that the behavior of RNN-based PMI and PHSIC are considerably different. Furthermore, interestingly, the behavior of PHSICs using different kernels is also different. Possible reasons for these observations are as follows: (1) the difference in the dependence measures (MI or HSIC) on which each model is based; (2) the validity or numerical stability of estimating PMI with RNN language models; and (3) differences in the behavior of PHSIC originating from differences in the plugged in kernels. A more detailed analysis of the compatibility between tasks and measures (or kernels) is attractive future work.