Logic-driven Indirect Supervision: An Application to Crisis Counseling

Mattia Medina Grespan[1], **Meghan Broadbent**[2], **Xinyao Zhang**[2], **Katherine E. Axford**[2], **Brent Kious**[3], **Zac Imel**[2] and **Vivek Srikumar**[1]
[1]Kahlert School of Computing, University of Utah
[2]Department of Educational Psychology, University of Utah
[3]Huntsman Mental Health Institute, Department of Psychiatry, University of Utah
mattiamg@cs.utah.edu

Abstract

Ensuring the effectiveness of text-based crisis counseling requires observing ongoing conversations and providing feedback, both labor-intensive tasks. Automatic analysis of conversations—at the full chat and utterance levels—may help support counselors and provide better care. While some session-level training data (e.g., rating of patient risk) is often available from counselors, labeling utterances requires expensive post hoc annotation. But the latter can not only provide insights about conversation dynamics, but can also serve to support quality assurance efforts for counselors. In this paper, we examine if inexpensive—and potentially noisy—session-level annotation can help improve label utterances. To this end, we propose a logic-based indirect supervision approach that exploits declaratively stated structural dependencies between both levels of annotation to improve utterance modeling. We show that adding these rules gives an improvement of 3.5% f-score over a strong multi-task baseline for utterance-level predictions. We demonstrate via ablation studies how indirect supervision via logic rules also improves the consistency and robustness of the system.

***Trigger warning**: This paper discusses suicide in the context of crisis counseling and includes examples illustrating such conversations.*

1 Introduction

Text-based crisis counseling services like Crisis Text Line[1] and the 988 Suicide & Crisis Lifeline[2] are increasingly adopted by people seeking confidential mental health support. They help thousands of texters every day. But the volume of users challenges the ability of crisis systems to provide consistently high-quality service. For example, in our experience with the regional suicide hotline, SafeUT,[3] we found that counselors can have extended shifts involving up to eight conversations with potentially suicidal clients *simultaneously*! Figure 1a shows an illustrative anonymized example of such a session.

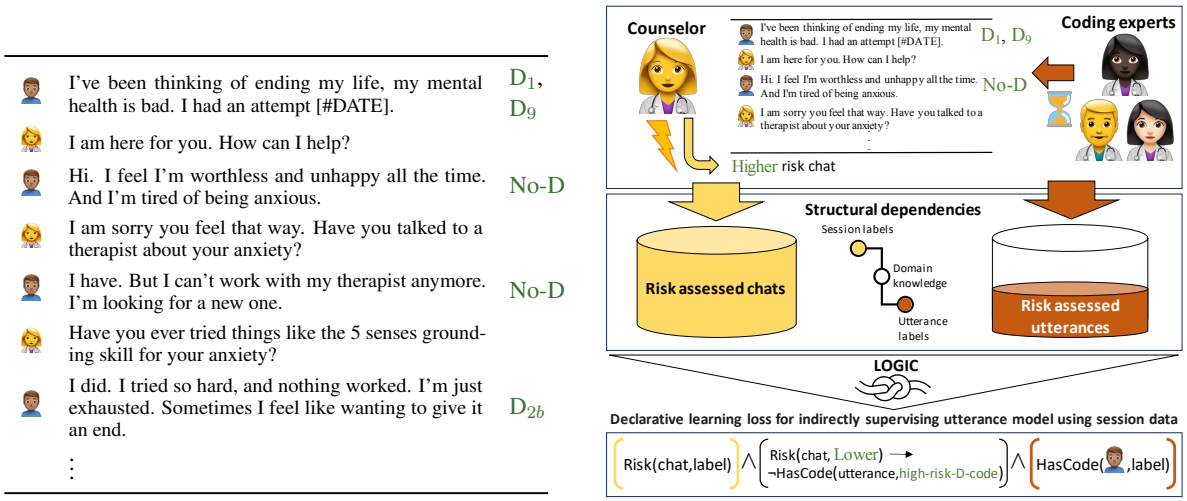
Addressing the twin problems of managing counselor workload and ensuring quality requires training new counselors and providing feedback to existing ones. In particular, understanding suicide risk in client utterances may help counselors learn to prioritize high-risk client situations, especially when dealing with multiple chats simultaneously or when fatigued. As Imel et al. (2017) note, scaling such efforts requires technological assistance. Previous work (e.g., Broadbent et al., 2023; Guzman-Nateras et al., 2022; Shrestha et al., 2021; Haque et al., 2020) has shown that NLP models can reliably assess risk in crisis chats. Yet, building models for risk assessment at the utterance level is challenging because of the dearth of training data.

Utterance-level risk labeling requires post hoc annotation by experts who follow a coding manual; the process can be slow and expensive. In contrast, session-level risk data is relatively easier to obtain. At the end of a session, in their standard workflow, counselors can tag the risk level (e.g., low- or high-risk) for record keeping requirements. Session-level assessments are undeniably useful (Xu et al., 2021; Bantilan et al., 2021); but the nuances of moment-to-moment situational judgments are also key for clinical training and supervision.

In this paper, we ask: *Can the easy-to-obtain session-level risk data help improve utterance risk classifiers?* These two tasks have structural dependencies between them: session-level classification of risk should be dependent on utterance-level classification, such that a session containing any high-risk utterances should be deemed high risk. This connection paves the way to extract auxiliary signal

[1]https://www.crisistextline.org
[2]https://988lifeline.org/
[3]https://safeut.org. This work was conducted under IRB oversight. Appendix A has more details.

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 11704–11722
July 9-14, 2023



(a) (b)

Figure 1: (a) Example snippet of an anonymized Higher risk chat session with associated utterance-level labels for client ( ) utterances. (b) Overview of the problem and our approach. We have two disjoint datasets of annotated crisis sessions: one set is labeled at the session level (with Higher or Lower risk) by the counselor immediately after the chat ends, the other set is labeled post hoc at the utterance level (with one or more risk status codes shown in Table 1) by coding experts. There are structural dependencies between the two levels of annotation. Our work proposes a framework that exploits these dependencies via a logic-guided declarative learning loss where the easy-to-obtain session data provides auxiliary supervision for the low-resource utterance classification task.

from the easily obtained session labels to indirectly supervise utterance models.

Prior work on indirect supervision with structured prediction (Chang et al., 2010a,b) focuses on feature-rich linear models. However, importing these ideas to the deep learning realm can be computationally untenable due to the discrete optimization step within the training loop. We propose a mechanism to instantiate this cross-task indirect supervision problem as a declarative learning objective encoded by logical constraints. For efficient training, these constraints are relaxed into differentiable losses (e.g., Richardson et al., 2022; Li et al., 2019; Rocktäschel et al., 2015). Figure 1b illustrates the approach. The flexibility of the framework allows us to incorporate further structural constraints inherent in the utterance-level task.

We show that the auxiliary supervision via constraints significantly improves utterance risk prediction over both direct supervision and strong multi-task baselines. Our analysis reveals that the rules also improve model consistency and robustness.

In summary, our contributions are: We introduce a framework for indirect supervision that uses relaxed logic. We instantiate it to the problem of using cheap, abundant, but noisy annotation (session-level risk labels) as auxiliary signal to improve the performance on a low-resource task (utterance-level labels). We show that structural dependencies across tasks help outperform a directly supervised and a strong multi-task baselines.

2 Crisis Counseling and Coding

In text-based crisis intervention, a client starts a chat session (also called an encounter) by typing a message, and the first available counselor replies to it. The session goes on till either the client finishes the conversation, or a certain amount of time elapses with no client response.

The volume of messages to text-based crisis services presents quality assurance challenges and demands increased counselor training. NLP-based tools can help both with quality control and for counselor feedback during training (Sharma et al., 2021; Demasi et al., 2020, 2019; Dinakar et al., 2015). In particular, monitoring chat-level and also utterance (or message) suicide risk status can be critical to improve service effectiveness. To build such models, we need risk assessment annotation at two levels: at the session level and the utterance level. The former is easy to obtain, whereas the latter is not.

Once a session concludes, counselors tag the conversation as being higher or lower suicide risk as part of their routine reporting requirements. Consequently, we can organically obtain session-level

Code	Notation
Lifetime ideation	D_1
Current ideation	D_2
Imminent risk	D_{2a}
Passive ideation	D_{2b}
Attempt in progress	D_3
Method chosen or considered	D_6
Availability of means	D_{6b}
Prior attempt(s)	D_9

Table 1: Client utterance risk status codes from the crisis chat scheme of Lake et al. (2022). Codes in **bold** are associated with client high suicide risk. We denote with No-D when there is no D-code label associated.

annotation, but perhaps with some noise due to provider fatigue.

In contrast, labeling the suicide risk status of client utterances needs careful post hoc analysis over the session. For this process, a group of expert annotators label each utterance using a standard coding system for risk. In this work, we use the crisis chat coding scheme of Lake et al. (2022).[4] Specifically, the coders assign to each client utterance zero, one, or more suicide risk status codes from the section D of the coding manual. We refer to these labels as "D-codes" in this paper; Table 1 lists the eight codes used. Utterances for which no codes apply are labeled with a special No-D label. The right columns of the example chats in Figure 1a and Appendix B show the D-codes associated with the client utterances. The subtle differences between the label definitions make the utterance-level annotation an expensive and slow process. Consequently, only a limited amount of labeled data with utterance suicide risk status is available.

With these two types of counseling annotation—the cheap and noisy session-level data, and the expensive and slow utterance-level data—we seek to use the naturally occurring session risk assessment signal to improve an utterance risk status model.

Datasets. We use two datasets from the regional suicide crisis hotline SafeUT. Both contain encounters consisting of text messages between the client and possibly multiple counselors. Since they were created in different development stages of SafeUT, they are disjoint: one with client utterances labeled for risk status and the other with labeled encounters. No encounter is annotated at both levels.

The first dataset, denoted as U, contains 425 sessions labeled by seven annotators: six graduate students and a psychology professor. The average session has 23 utterances, with 13 from the client. Each annotator independently labeled client utterances with a nine-dimensional label indicating a no-code or a *combination* of risk status D-codes (Table 1) and achieved a high 0.8 intra-class correlation coefficient. The set U contains 4912 client utterances, 688 of which are labeled with at least one D-code. The second dataset, denoted as E, contains 5990 encounters labeled by trained SafeUT counselors with binary risk assessment labels. They labeled 879 and 5111 encounters with higher and lower risk respectively.[5]

Problem statement. Previous research with similar kinds of data has used multi-task learning techniques to create joint representations of the input, successfully improving utterance-level prediction (Gibson et al., 2022; Cahn, 2021).

The two types of labels are tied by structural dependencies. From the definitions of the D-codes, and their associated coding manual (Lake et al., 2022), we observe that certain utterance D-codes suggest a higher risk session: an encounter containing a client utterance coded with Imminent risk (D_{2a}), Attempt in progress (D_3), Method chosen or considered (D_6) or Availability of means (D_{6b}) must be assessed as having higher risk. Conversely, a lower risk assessed encounter cannot contain an utterance coded with any high-risk D-code.

Beyond the cross-task dependencies, the definition of the D-codes also entails that the occurrence of certain labels logically necessitates the occurrence of certain others. For example, a client who has attempted suicide in the past had (at least) one lifetime suicide ideation. Hence, an utterance coded with Prior attempt(s) (D_9) must also be coded with Lifetime ideation (D_1).[6] We list all the D-code dependencies in the appendix (Table 13).

The structural dependencies between the tasks open the possibility of using encounter-level annotation as indirect supervision for utterance-level risk status coding. Moreover, the dependencies between D-codes can be used to guide models to-

[4]Our SafeUT data uses a revised version of the Suicide Risk Factors in the Crisis Chat Transcript Abstraction.

[5]The set E was previously used for session binary risk assessment by Broadbent et al. (2023).

[6]This dependency between the D_9 and D_1 labels is from the D-code annotation guidelines. However, we note that it is theoretically possible that one could make a suicide attempt without ideation. Indeed, since we are dealing with mental states of people, the dependencies between the D-codes are actually only highly probable rather than being inviolable mandates. But rule violations are psychologically improbable; so for this work, we can treat them as constraints.

wards more consistent and robust utterance risk status prediction despite the paucity of data.

We ask: *Can we exploit the structural dependencies between the two kinds of annotation and within the D-codes to aid utterance-level prediction?*

3 Indirect Supervision via Logic

In this work, we introduce a logic-guided indirect supervision framework that uses cross-task dependencies to transfer signal from the session data to the utterance models. The declarative nature of the structural dependencies between the two tasks allows us to express them as predicate logic rules.

The question of indirect supervision with structural constraints has been studied in the structured prediction literature (Chang et al., 2010a,b). However, instantiating these approaches for neural networks is computationally expensive because of the need to perform combinatorial inference in the innermost loop of the already slow training process.

Instead, we build on the approach presented in Li et al. (2019) and Medina Grespan et al. (2021) and relax the rules to define sub-differentiable losses that encourage utterance and session models to satisfy them. Doing so allows us to train a jointly constrained pair of models from the two data sources.

The rest of this section expands on this intuition to present a declarative formulation of the problem. The next section focuses on using the formulation to design a loss function for learning.

3.1 Notation

We denote by $e = \{m_1, m_2, \ldots, m_n\}$ an encounter with n utterances where each m_i represents a client or a counselor utterance. We denote by $\mathcal{R} = \{\text{Lower}, \text{Higher}\}$ the set of risk labels at the session level, and by $\mathcal{D}$ the set of all risk status utterance labels in Table 1. Additionally, we denote the no-code label as No-D. We denote the subset of high-risk D-codes (bold rows in Table 1) by $\mathcal{H}$.

We represent the fact that an encounter e has risk $r \in \mathcal{R}$ as the predicate $\texttt{Risk}(e, r)$. Similarly, we define the predicates $\texttt{HasCode}(m, d)$ to denote the fact that an utterance m has the label $d \in \mathcal{D}$, and $\texttt{NoCode}(m)$ to denote that the label of m is No-D.

3.2 Declarative Problem Formulation

For the declarative loss learning approach, we first need to represent the labeled data and structural constraints in predicate logic.

Data Constraints. The dataset of encounters E sets the Risk for each session it contains:

$$\forall (e, r) \in E, \texttt{Risk}(e, r). \tag{1}$$

To represent the fact that a client utterance m is labeled with a set of D-codes $\mathcal{D}^* \subset \mathcal{D}$, we need to ensure that (a) the labels of m are in $\mathcal{D}^*$, and (b) neither the No-D, nor other D-codes should apply for the message. For notational convenience, we will call these $\texttt{M}_1$ and $\texttt{M}_2$ respectively.

$$\texttt{M}_1(m, \mathcal{D}^*) := \bigwedge_{d \in \mathcal{D}^*} \texttt{HasCode}(m, d) \tag{2}$$

$$\texttt{M}_2(m, \mathcal{D}^*) := \neg\texttt{NoCode}(m) \wedge \bigwedge_{d \in \mathcal{D} \setminus \mathcal{D}^*} \neg\texttt{HasCode}(m, d) \tag{3}$$

Using these helper predicates, we can represent a session in the utterance labeled data U. Each client utterance in a session $e \in U$ either has a set of D codes associated with it, or has the No-D label.

$$\forall e \in U,\ \forall (m, \mathcal{D}^*) \in e,\ \texttt{M}_1(m, \mathcal{D}^*) \wedge \texttt{M}_2(m, \mathcal{D}^*), \quad \forall (m, \text{No-D}) \in e,\ \texttt{NoCode}(m). \tag{4}$$

Joint constraint. A session assessed with Lower risk must not contain a client utterance with a high-risk D-code from the set $\mathcal{H}$. This constraint applies for every utterance in the session. Importantly, the rule applies to all sessions, whether they are labeled or not, and in particular, to sessions in both datasets E and U. We can write:

$$\forall e \in E \cup U,\ \forall m \in e,\ \forall d \in \mathcal{H} \quad \texttt{Risk}(e, \text{Lower}) \rightarrow \neg\texttt{HasCode}(m, d). \tag{5}$$

D-Code constraints. For a set of pairs of D-codes (d_i, d_j), if the former applies to a message, so should the latter. We will refer to the full set of pairs (Table 13 in the appendix) as RULES. These label dependencies apply to every message in every encounter in both datasets E and U. We can write:

$$\forall e \in E \cup U,\ \forall m \in e,\ \forall (d_i, d_j) \in \text{RULES}, \quad \texttt{HasCode}(m, d_i) \rightarrow \texttt{HasCode}(m, d_j). \tag{6}$$

NoCode constraint. Our final constraint enforces structural consistency among the utterance risk predictions. In the multi-label setting, every utterance either has the No-D label or a combination labels in $\mathcal{D}$, but never both. The constraint holds for all encounters in our data. We write:

$$\forall e \in E \cup U, \forall m \in e, \quad \texttt{NoCode}(m) \leftrightarrow \bigwedge_{d \in D} \neg\texttt{HasCode}(m, d). \tag{7}$$

Full declarative specification. We can state the desired properties involving our predicates as a formula composed by the conjunction of the expressions (1), (4)—representing the labeled datasets—and the expressions (5), (6), and (7)—representing the domain knowledge rules. Together, these can be thought of as "contracts" that any model for the tasks should seek to satisfy.

4 From Logic to Losses

In our declarative formulation, we have three atomic predicates: `HasCode`, `NoCode` and `Risk`. We model the truth value of these predicates as the output probabilities of a transformer-based classifier. We denote the relaxed truth value of the predicate classifiers with square brackets. For instance, given a session e, we denote the predicted probability that the fact $\texttt{Risk}(e, \text{Higher})$ holds as $[\texttt{Risk}(e, \text{Higher})]$.

All the constraints we have encountered will be relaxed into differentiable forms, such that the truth values of the atomic predicates define the truth value of the entire loss under the relaxation. Consequently, learning the three predicates will require optimizing their parameters to maximize the truth value of the relaxed declarative loss.

4.1 Multi-task Predicate Models

We use a joint neural model for the relaxed truth values of the predicates `NoCode`, `HasCode` and `Risk`. The network receives an input session and predicts the probabilities of risk for the entire session, and client risk status for each utterance.

Our models are based on RoBERTa (Liu et al., 2019). To make the embeddings domain-aware, following Gururangan et al. (2020), we adapted the RoBERTa-base model using a large corpus of 2 million fully unlabeled `SafeUT` utterances.

Given a session, we obtain representations for each utterance by averaging its token RoBERTa embeddings. We input the utterance representations into a 2-layer transformer encoder to obtain session-contextualized utterance embeddings. The average of the utterance embeddings is used to represent the entire session. The session embedding is the input of a linear layer with two outputs, whose softmaxed values serve as the Lower and Higher risk probabilities of the session. These probabilities model the truth values of the `Risk` predicate.

To each utterance embedding in the session, we apply a linear layer with $|\mathcal{D}| + 1$ outputs followed by an element-wise sigmoid activation. These give us the utterance risk status probabilities and the No-D probability, which model the truth value of the `HasCode` and `NoCode` predicates.

Appendix C gives additional details about the model architecture. Note that since the output probabilities share a common session-contextualized embedding model, they represent a simple multi-task model where each one task has the opportunity to influence and improve the other.

4.2 Losses

The key idea behind our relaxation approach is that each boolean operator can be softened into a sub-differentiable function. We follow the recommendations of Medina Grespan et al. (2021) and use the $\mathcal{R}$-product t-norm relaxations of the logic operators to produce loss functions. Table 11 in the appendix shows the relaxations for each operator.

Applying the relaxation to rules in section 3.2, we can construct loss functions that we then optimize. In other words, every loss defined below has an analogue in section 3.2.

Data losses. The expression (1) requires all the predicates representing the labeled sessions in E should hold. This is equivalent to asking the conjunction of $\texttt{Risk}(e, r)$ facts for all (e, r) pairs in E to hold, which is relaxed as the product of its conjuncts. Equivalently, we can minimize the negative log the expression, and recover the standard cross-entropy loss for encounter risk classification.

$$L_E = \sum_{(e,r)\in E} -\log\left[\texttt{Risk}(e, r)\right] \tag{8}$$

Analogously, we can write the losses for the helper predicates in expressions (2) and (3),

$$\ell_{\texttt{M}_1}(m, \mathcal{D}^*) = \sum_{d\in\mathcal{D}^*} -\log[\texttt{HasCode}(m, d)]$$

$$\ell_{\texttt{M}_2}(m, \mathcal{D}^*) = \log(1 - [\texttt{NoCode}(m)]) + \sum_{d\in\mathcal{D}\setminus\mathcal{D}^*} \log(1 - [\texttt{HasCode}(m, d)])$$

These helper losses let us write the loss of the utterance labeled data U, thus relaxing the Boolean expression (4) to recover the binary cross entropy loss for multi-label classification:

$$L_U = \sum_{e\in U} \Big(\sum_{(m,\mathcal{D}^*)\in e} \big(\ell_{\texttt{M}_1}(m, \mathcal{D}^*) + \ell_{\texttt{M}_2}(m, \mathcal{D}^*)\big) + \sum_{(m,\text{No-D})\in e} -\log[\texttt{NoCode}(m)] \Big) \tag{9}$$

Joint constraint loss. For the joint constraint (5), using the $\mathcal{R}$-Product definition of implication, we obtain a loss composed of the sum of ReLU functions:

$$L_{\text{Joint}} = \sum_{e \in U \cup E} \sum_{m \in e} \sum_{d \in \mathcal{H}} \ell_{\text{J}}(e, m, d) \quad (10)$$

where,

$$\ell_{\text{J}}(e, m, d) = \text{ReLU}\Big(\log[\texttt{Risk}(e, \text{Lower})] - \log(1 - [\texttt{HasCode}(m, d)]) \Big) \quad (11)$$

D-Code constraints loss. In a similar fashion as above, we can derive the D-code dependencies (6).

$$L_{\text{D}} = \sum_{e \in U \cup E} \sum_{m \in e} \sum_{(d_i, d_j) \in \text{RULES}} \ell_{\text{I}}(m, d_i, d_j) \quad (12)$$

where,

$$\ell_{\text{I}}(m, d_i, d_j) = \text{ReLU}\Big(\log[\texttt{HasCode}(m, d_i)] - \log[\texttt{HasCode}(m, d_j)] \Big)$$

NoCode constraint loss. Following the structure of the NoCode constraint (7), we can write the NoCode loss as

$$L_{\text{NoCode}} = \sum_{e \in U \cup E} \sum_{m \in e} \ell_n(m, d) \quad (13)$$

However, unlike the cases we have seen so far, naively applying the conversion rules gives us a loss that is not stable for learning. This was also observed by Li et al. (2020), who suggest that for stability, the conjunction of the negation on the right-hand side of the double implication be relaxed using the Gödel conjunction (which is the min of the conjuncts). Doing so and simplifying gives us:

$$\ell_n(m, d) = \Big| \log\left([\texttt{NoCode}(m)]\right) - \log\left(1 - \max_{d \in \mathcal{D}} [\texttt{HasCode}(m, d)]\right) \Big| \quad (14)$$

Full logic-based loss. Just as the full declarative specification is the conjunction of individual components, the problem of learning the predicate models requires minimizing the total loss:

$$L = L_U + \lambda_E L_E + \lambda_{\text{NoCode}} L_{\text{NoCode}} + \lambda_{\text{D}} L_{\text{D}} + \lambda_{\text{Joint}} L_{\text{Joint}} \quad (15)$$

	No-D	$\mathcal{D}$	Size
Train	1796	231	135
Dev.	1732	193	144
Test	1384	264	146

Table 2: Data statistics of the utterance risk set U. The first and second columns show the number of client utterances labeled with no-code and D-codes respectively, and the third column the number of encounters.

	Lower	Higher	Size
Train	4600	793	5393
Test	511	86	597

Table 3: Data statistics of the encounter set E. The first and second columns show the number of encounters labeled with Lower and Higher risk respectively, and the last column shows the number of encounters.

Here, the λ's are non-negative hyper-parameters that regulate the signal from each loss term. Importantly, the unsupervised losses L_{Joint}, L_{D} and L_{NoCode} apply to encounters in both datasets E and U; they are not defined over ground truth labels. The joint loss serves to transfer signal from the encounter data to the utterance predictors, while the other two unsupervised losses enforce structural consistency in the utterance predictors.

5 Experiments and Results

5.1 Experimental Setup

Data. We partition the utterance-level dataset U with stratified splits of 135, 144 and 146 encounters for training, development and testing respectively. We split the encounter-level dataset E into 5,393 encounters for training and 597 encounters for testing. Tables 2 and 3 provide summary statistics.

Baselines. Our proposed approach optimizes the total loss that includes all the relaxed rule components. We compare our system against two baselines with the same architecture but simplified rule-less losses: $L_{\text{baseline}} = L_U$ and $L_{\text{multi-task}} = L_U + L_E$. The first baseline is trained only on utterance D-coded data. The second baseline incorporates the labeled session-level data under a standard multi-task learning regime that shares representations.

Training details. We train all models on the U and E training splits. For each training epoch, we use a random combination of batches from U and E—respectively computing the truth values of the

	F_1	P	R
Baseline	43.2(4.2)	**57.1**(7.2)	34.7(3.4)
Multi-Task (MT)	46.5(3.3)	54.9(5.0)	40.8(5.0)
MT+Rules	**50.0**(0.7)	49.4(3.6)	**51.0**(3.6)

Table 4: Utterance code multi-label classification F_1, precision (P) and recall (R) micro average scores.

	F_1	P	R
Baseline	13.7(13.0)	10.6(10.8)	24.8(25.0)
Multi-Task (MT)	**50.6**(3.9)	**43.7**(3.4)	60.9(8.8)
MT+Rules	47.5(0.8)	33.0(0.7)	**84.4**(3.1)

Table 5: Risk assessment binary classification F_1, precision (P) and recall (R) micro average scores.

	NoCode	D-Code	Joint
Baseline	41.4(10.6)	2.2(2.8)	30.8(18.6)
Multi-Task (MT)	74.6(55.6)	1.9(1.2)	11.6(8.4)
MT+Rules	27.2(5.6)	0.0(0.0)	0.6(0.9)

Table 6: Number of utterances in the test split of the utterance labeled data U violating each of the constraints.

predicates `HasCode` and `NoCode`, and `Risk`. Since our goal is to build a better utterance predictor, we use the development set from U for hyperparameter tuning and model selection using the micro-average of the F1 score in multi-label utterance classification. We train the models for 150 epochs with early stopping after 50 epochs using AdamW optimizer (Loshchilov and Hutter, 2019). We refer the reader to appendix C.4 for details.

Evaluation. For the utterance and chat labels, we report the precision, recall and F_1 micro-averages. Further, we measure the consistency of model predictions by analyzing how much they violate the declarative rules we are incorporating. We report the average performance of the models on the test splits across five different training random seeds.

5.2 Main Results

Utterance results. Table 4 reports the utterance D-code classification results over utterances labeled with at least one D-code.

We expect the baseline to already have some domain tuning because the RoBERTa embeddings were additionally pre-trained on counseling text. Standard multi-task (MT) classification improves the F_1 score by 3.3% with respect to the baseline, corresponding to a 6.1% increase in recall. We can attribute this improvement to the shared feature space in the transformer encoder layers becoming better from the encounter labeled data. Finally, we observe that introducing the relaxed rules loss components (MT+Rules) produces a F_1 gain of 3.5% over the already improved multi-task system (corresponding to a 10.2% improvement in recall). Each subsequent F_1 improvement is statistically significant at $p < 0.05$ using the paired t-test. Related to the recall increase, we observe that the F_1 for the majority label No-D dropped. Compared to the baseline's 95.2%, the full and multi-task systems' scores dropped to 91.5% and 88.9% respectively. Importantly, in this domain, the recall improvements are desired. False positive D-code predictions are preferable to missing any important suicide-related cues.

Session results. Table 5 reports F_1, precision and recall scores for the Higher risk label.

The baseline is unsurprisingly as good as random; it does not have any access to session-level risk supervision. Compared to the multi-task baseline, we observe a drop in F_1 performance in our system. We discover that this difference corresponds to a 10.7% drop in precision, but also to a significant gain of 23.5% in recall. These results show that incorporating indirect signal from the rules prioritizes recall which aligns with the goals of suicide risk detection application: Improved recall for the Higher risk label can help focus counselors attention to such clients.[7]

Constraint violations. Table 6 shows how often (on average across random seeds) the systems violate each of the declarative rules.

For the NoCode constraint, which introduces a mutual exclusion between the No-D label and any D-code for every utterance, we find that the multi-task system has more violations than the baseline. This implies that multi-task model's gain in utterance D-code recall over the baseline (Table 4) is related to errors where the system assigns utterances with the right D-codes but also the No-D label. Adding the rules mitigates this problem.

For the D-Code rules, which enforce dependencies between D-codes, even the baselines have only

[7] In this work, we do not consider encounter-only model training. Broadbent et al. (2023) showed that doing so—i.e., optimizing the loss L_E alone—results in better session-level risk classification. Our focus here is the D-codes, and we tune our models and hyper-parameters for utterance-level predictions. However, we also note that our models, jointly trained with rules, improve recall over their reported performance.

	F_1	P	R
Multi-Task (MT)	46.5(3.3)	54.9(5.0)	40.8(5.0)
MT+NoCode Rule	47.7(1.2)	45.0(1.5)	50.9(3.4)
MT+D-Code Rules	45.4(2.9)	55.8(6.8)	38.7(4.0)
MT+Joint Rule	49.7(2.8)	56.8(7.7)	44.6(3.0)

Table 7: Ablation results on D-code prediction.

	F_1	P	R
Multi-Task (MT)	50.6(3.9)	43.7(3.4)	60.9(8.8)
MT+NoCode Rule	45.8(2.0)	31.4(2.1)	85.3(3.9)
MT+D-Code Rules	45.2(2.0)	31.5(2.4)	80.7(4.8)
MT+Joint Rule	48.8(3.7)	36.2(6.0)	77.7(7.6)

Table 8: Ablation results on session risk prediction.

few violations. Nevertheless, our system recovers perfect consistency with respect to these rules.

Lastly, the joint rule prohibits all client utterances from Lower risk encounters from having any high-risk D-code. Given that we have a random risk classifier in the baseline, we only compare system violation performance against the multi-task system for this rule. We observe that our system improves in terms of violations for the joint rule implying that it successfully incorporates the knowledge from the L_{Joint} loss during training.

5.3 Ablation Analysis

To better understand the impact of each rule, we perform an ablation study with respect to the multi-task baseline. Tables 7, 8 and 9 report the impact of each rule individually added during training.

Adding only NoCode rule. As expected, we see that NoCode rule violations drop when adding only the NoCode rule loss (Table 9). Furthermore, the NoCode rule loss by itself improves utterance F_1 (1.2%) by reducing the precision and increasing the recall by 10% each (Table 7). This improvement indicates that the system is predicting more D-codes and fewer No-D. (As expected, the F_1 score on the No-D label decreases from 91.5% for the multi-task baseline to 87.5%.)

We observe a more dramatic effect on the encounter risk classifier with a big improvement in recall at the cost of a significant drop in precision, resulting in an overall F_1 drop of 4.8% (Table 8). In this case, updating the model weights to optimize the NoCode loss (L_{NoCode}) defined at the utterance level makes the encounter-level risk assessment classifier to predict more Higher risk.

	NoCode	D-Code	Joint
Multi-Task (MT)	74.6(55.6)	1.9(1.2)	11.6(8.4)
MT+NoCode Rule	9.4(3.4)	0.0(0.0)	2.0(1.6)
MT+D-Code Rules	118.2(81.3)	0.0(0.0	9.6(7.4)
MT+Joint Rule	68.4(5.7)	1.8(2.0)	8.4(4.7)

Table 9: Ablation rules on utterance rule violations

Adding only D-Code rules. For D-code classification, precision increases at the cost of recall (Table 7). In this case, the system incorrectly predicts messages without any label (No-D and D-codes) to trivially satisfy all the D-Code rules; hence, the reduced D-code recall.

The system has perfect consistency for the D-Code rules as expected. Analyzing the effect of the D-Code loss on the risk classifier, we observe a similar behaviour as using only the NoCode loss. This similarity implies that adding constraints at the utterance level affects the weights in the shared feature space to make the risk classifier more sensitive to risk, i.e. more recall at the cost of precision.

Adding only the joint rule. We observe a significant 3.2% gain in F_1 performance corresponding to precision and recall gains of 1.9% and 3.8% respectively (Table 7). We attribute this improvement to the indirect supervision coming from the risk classifier through the inter-label dependency encoded by the (relaxed) joint constraint.

Analyzing the performance on the risk classifier we observe a comparable F_1 performance with respect to the multi-task baseline with a considerable 7.5% drop in precision offsetting a significant 17% gain in recall (Table 8). In this case, the signal from the utterance risk classifier on high-risk D-codes makes the encounter risk assessment model more sensitive to risk, which is a desirable behaviour. The classifier using only the joint rule loss, unsurprisingly, does not improve NoCode and D-Code rules violations as they are not part of the objective function during training, but it improves the joint rule violations (Table 9).

6 Error Analysis

We manually examined false positive and false negative predictions of the MT+Rules model on the development split of U. For this analysis, we used the model corresponding to the random seed that provided the best micro F1 performance on Table 4. We found four dominant kinds of errors, listed below.

Passive [D_{2b}] vs Current [D_2] Ideation. Confusion between passive and current ideation accounts for 27% of the total errors. We observe that half of these mistakes are edge cases which can be hard to discern even for a human. For example, the D_{2b} utterance "I am having those thoughts again. Being better off dead" is classified with both D_{2b} and D_2.

Lifetime [D_1] vs Current [D_2] Ideation. The inability to distinguish lifetime and current suicidal ideation (perhaps related to deficiencies in temporal reasoning) accounts for 15% of the errors. For example, the D_2 utterance "I'm worried. She has sent me a text saying she was going to commit suicide" is classified with both D_2 and D_1.

Excessive No-code [No-D]. Missing D-codes account for 20% of the errors. We observe that in almost all of this cases, the true label depends on previous context (e.g., "Yes...","Not really").

Commonsense Knowledge. We observe that 4% of the errors come from poor commonsense reasoning. For example, our model does not predict D_2 and D_{2b} for the utterances "a kid on social media posted bloody cuts, the caption said bye bye!...", and "I know I am only alive for my friends and food! LOL" respectively.

The table in Appendix E shows additional examples of these errors.

7 Related Work & Discussion

Mental Health NLP-based methods have proven useful to detect risk in mental health counseling. Benton et al. (2017) built a multi-task model to predict suicide risk on social media, and report improvements over single-task models trained on limited data. Gibson et al. (2022) developed a multi-task model to predict therapist use of psychological interventions for each talk turn. Like our multi-task model, their model simultaneously learned two different labeling schemes by building two separate encoders for respective tasks plus a shared encoder. Their multi-task model also outperformed single-task models. Our work goes beyond the multi-task approach and incorporates indirect supervision from structural dependencies between the two sources of annotation.

Indirect supervision. Our work is conceptually related to an indirect supervision joint inference paradigm (e.g., Roth, 2017) which leverages domain knowledge to enforce structural dependency constraints. Other efforts also use indirect supervision paradigms (as presented in Wang and Poon (2018)) for biomedical and mental health domains. Cusick et al. (2021) use weak supervision from a regular-expression-based algorithm that successfully leverages noisy labels that improve suicidal ideation on clinical notes classification. Fu et al. (2021) use a suicide ontology-based knowledge graph for distant supervision in suicide risk detection on social comments.

Logic-driven learning. Among a variety of logic-driven learning approaches (e.g., Besold et al., 2017), our method is probably the closest to probabilistic soft logic of Kimmig et al. (2012). This approach softens booleans to the interval $[0, 1]$ using the Lukasiewicz t-norm relaxation. This approach has shown promising results in several empirical studies, especially in low-data regimes. For example, Li et al. (2019) used the product t-norm to relax logic for entailment, while Wang et al. (2020) showed that by introducing logic constraints, their model outperformed benchmark models on the event-event relation data that lacked joint labels. Our work shows that the logic-driven learning framework can be used to transfer supervision signal between tasks with very different input (encounter vs. session) and output (binary vs. multi-label classification) characteristics.

8 Conclusion

In this work, we study the problem of predicting utterance-level labels in a suicide crisis chat with the goal of better understanding such sessions and providing better feedback to fatigued counselors. We propose a fully declarative framework that integrates different data sources with a logic-guided loss. We experiment with two text-based crisis counseling datasets from the same source, but with different and disjoint annotations. One level of annotation—the session level—occurs naturally but is noisy, while the other level of annotation—the utterance level—is expensive but precise. Our results show that exploiting the structural dependencies among the sources of annotations allows the session labels to help improve the utterance model.

9 Limitations

Our experiments reveal that simultaneously incorporating more rules into the loss produces better

performance in the task of interest (Table 4). These results indicate that rules working in tandem significantly complement supervision coming from both sources of direct annotation under a fully declarative loss. Nevertheless, controlling the influence of each term in the loss is crucial for training stability. We found that the system has different sensitivities to each term in the loss, requiring a full search over the λ hyper-parameters (15). From this perspective, the possible benefits of increasing the number of rules in the loss come at the cost of more difficult learning.

Due to hardware limitations of the protected environment server that stores the datasets we use, RoBERTa-base was the best model that could fit in the available GPUs. Although other pre-trained embeddings could provide better performance, we argue that this is orthogonal to our contribution of incorporating indirect supervision under a fully declarative learning framework. Moreover, integrating logic-driven frameworks and prompt-based models like T5 is an interesting future line of work.

Choosing RoBERTa as the underlying embedding foundation of our system introduces all the inherent limitations of large language models (Bender et al., 2021). From this standpoint, we envision the application of these sorts of systems as a human-guided tool used only for counselor training and quality assurance, and never for real counseling sessions.

10 Ethics Statement

Hovy and Spruit (2016) list several ethical issues in the study and application of natural language processing, and advocate increased awareness of possible adverse social impacts. This is especially true in mental health care in general, and in crisis services in particular. Linthicum et al. (2019) points out the latent bias in the demographic composition of a dataset, with the potential risk of excluding underrepresented populations. In addition, machines cannot understand the social meanings of some biased datapoints, such as particular language use that could be inappropriate or offensive to particular cultural groups. When picking up these biases, the model may run the risk of reinforcing these prejudices if no manual check is available (Lin et al., 2022). This is true not only for patients but equally for clinicians. Although our model was designed with a clinical application in mind, with no access to the demographic information of patients or clinicians due to confidentiality concerns, the current model should not be interpreted as a system that can be applied directly to local crisis services without manual supervision. Instead, this study should be seen as a test for the feasibility of multi-task learning in a particular clinical setting. If the model is ultimately applied to crisis services, it still should not be allowed to run on its own or override manual judgment, but should instead be used as an assisting tool to better inform clinicians in their clinical cases or training.

11 Acknowledgments and COI

The authors acknowledge the support of the Utah State Board of Education SafeUT Research and Quality Improvement Program Grant. This material is based in part upon work supported by the National Science Foundation under Grant #1822877. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation. We also thank the members of the Utah NLP group, and Tao Li for their feedback on previous iterations of this work, and the ACL reviewers for their valuable feedback.

Conflict of interest disclosure. Zac Imel is a co-founder and minority shareholder in Lyssn.io, a technology company focused on developing tools to improve the quality of behavioral healthcare.

References

Niels Bantilan, Matteo Malgaroli, Bonnie Ray, and Thomas D. Hull. 2021. Just in time crisis response: suicide alert system for telemedicine psychotherapy settings. *Psychotherapy Research*, 31(3):289–299. PMID: 32558625.

Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '21, page 610–623, New York, NY, USA. Association for Computing Machinery.

Adrian Benton, Margaret Mitchell, and Dirk Hovy. 2017. Multitask learning for mental health conditions with limited social media data. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 152–162, Valencia, Spain. Association for Computational Linguistics.

Tarek R. Besold, Artur S. d'Avila Garcez, Sebastian Bader, Howard Bowman, Pedro M. Domingos, Pas-

cal Hitzler, Kai-Uwe Kühnberger, Luís C. Lamb, Daniel Lowd, Priscila Machado Vieira Lima, Leo de Penning, Gadi Pinkas, Hoifung Poon, and Gerson Zaverucha. 2017. Neural-symbolic learning and reasoning: A survey and interpretation. *CoRR*, abs/1711.03902.

Meghan Broadbent, Mattia Medina Grespan, Katherine Axford, Xinyao Zhang, Vivek Srikumar, Brent Kious, and Zac Imel. 2023. A machine learning approach to identifying suicide risk among text-based crisis counseling encounters. *Frontiers in Psychiatry*, 14.

Daniel Cahn. 2021. Deephelp: Deep learning for shout crisis text conversations.

Ming-Wei Chang, Dan Goldwasser, Dan Roth, and Vivek Srikumar. 2010a. Discriminative learning over constrained latent representations. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 429–437, Los Angeles, California. Association for Computational Linguistics.

Ming-Wei Chang, Vivek Srikumar, Dan Goldwasser, and Dan Roth. 2010b. Structured output learning with indirect supervision. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, page 199–206, Madison, WI, USA. Omnipress.

Marika Cusick, Prakash Adekkanattu, Thomas R. Campion, Evan T. Sholle, Annie Myers, Samprit Banerjee, George Alexopoulos, Yanshan Wang, and Jyotishman Pathak. 2021. Using weak supervision and deep learning to classify clinical notes for identification of current suicidal ideation. *Journal of Psychiatric Research*, 136:95–102.

Orianna Demasi, Marti A. Hearst, and Benjamin Recht. 2019. Towards augmenting crisis counselor training by improving message retrieval. In *Proceedings of the Sixth Workshop on Computational Linguistics and Clinical Psychology*, pages 1–11, Minneapolis, Minnesota. Association for Computational Linguistics.

Orianna Demasi, Yu Li, and Zhou Yu. 2020. A multi-persona chatbot for hotline counselor training. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3623–3636, Online. Association for Computational Linguistics.

Karthik Dinakar, Jackie Chen, Henry Lieberman, Rosalind Picard, and Robert Filbin. 2015. Mixed-initiative real-time topic modeling & visualization for crisis counseling. In *Proceedings of the 20th International Conference on Intelligent User Interfaces*, IUI '15, page 417–426, New York, NY, USA. Association for Computing Machinery.

Guanghui Fu, Changwei Song, Jianqiang Li, Yue Ma, Pan Chen, Ruiqian Wang, Bing Xiang Yang, and Zhisheng Huang. 2021. Distant supervision for mental health management in social media: Suicide risk classification system development study. *Journal of Medical Internet Research*, 23(8):e26119.

James Gibson, David C. Atkins, Torrey A. Creed, Zac Imel, Panayiotis Georgiou, and Shrikanth Narayanan. 2022. Multi-label multi-task deep learning for behavioral coding. *IEEE Transactions on Affective Computing*, 13(1):508–518.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for Computational Linguistics.

Luis Guzman-Nateras, Viet Lai, Amir Pouran Ben Veyseh, Franck Dernoncourt, and Thien Nguyen. 2022. Event detection for suicide understanding. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 1952–1961, Seattle, United States. Association for Computational Linguistics.

Farsheed Haque, Ragib Un Nur, Shaeekh Al Jahan, Zarar Mahmud, and Faisal Muhammad Shah. 2020. A transformer based approach to detect suicidal ideation using pre-trained language models. In *International Conference on Computer and Information Technology (ICCIT)*, pages 1–5.

Dirk Hovy and Shannon L. Spruit. 2016. The social impact of natural language processing. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 591–598, Berlin, Germany. Association for Computational Linguistics.

Zac E Imel, Derek D Caperton, Michael Tanana, and David C Atkins. 2017. Technology-enhanced human interaction in psychotherapy. *Journal of counseling psychology*, 64(4):385.

Angelika Kimmig, Stephen H. Bach, Matthias Broecheler, Bert Huang, and Lise Getoor. 2012. A short introduction to probabilistic soft logic. In *NIPS 2012*.

Alison M. Lake, Thomas Niederkrotenthaler, Rebecca Aspden, Marjorie Kleinman, Amanda M. Hoyte-Badu, Hanga Galfalvy, and Madelyn S. Gould. 2022. Lifeline crisis chat: Coding form development and findings on chatters' risk status and counselor behaviors. *Suicide and Life-Threatening Behavior*, 52(3):452–466.

Tao Li, Vivek Gupta, Maitrey Mehta, and Vivek Srikumar. 2019. A logic-driven framework for consistency of neural models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3924–3935, Hong Kong, China. Association for Computational Linguistics.

Tao Li, Parth Anand Jawale, Martha Palmer, and Vivek Srikumar. 2020. Structured tuning for semantic role labeling. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8402–8412, Online. Association for Computational Linguistics.

Inna Lin, Lucille Njoo, Anjalie Field, Ashish Sharma, Katharina Reinecke, Tim Althoff, and Yulia Tsvetkov. 2022. Gendered mental health stigma in masked language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 2152–2170, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Kathryn P. Linthicum, Katherine Musacchio Schafer, and Jessica D. Ribeiro. 2019. Machine learning in suicide science: Applications and ethics. *Behavioral Sciences & the Law*, 37(3):214–222.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Mattia Medina Grespan, Ashim Gupta, and Vivek Srikumar. 2021. Evaluating relaxations of logic for neural networks: A comprehensive study. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 2812–2818. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Kyle Richardson, Ronen Tamari, Oren Sultan, Dafna Shahaf, Reut Tsarfaty, and Ashish Sabharwal. 2022. Breakpoint transformers for modeling and tracking intermediate beliefs. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9703–9719, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Tim Rocktäschel, Sameer Singh, and Sebastian Riedel. 2015. Injecting logical background knowledge into embeddings for relation extraction. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1119–1129, Denver, Colorado. Association for Computational Linguistics.

Dan Roth. 2017. Incidental supervision: Moving beyond supervised learning. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, page 4885–4890. AAAI Press.

Ashish Sharma, Inna W. Lin, Adam S. Miner, David C. Atkins, and Tim Althoff. 2021. Towards facilitating empathic conversations in online mental health support: A reinforcement learning approach. In *Proceedings of the Web Conference 2021*, WWW '21, page 194–205, New York, NY, USA. Association for Computing Machinery.

Amendra Shrestha, Nazar Akrami, Lisa Kaati, Julia Kupper, and Matthew R. Schumacher. 2021. Words of suicide: Identifying suicidal risk in written communications. In *IEEE International Conference on Big Data (Big Data)*, pages 2144–2150.

Hai Wang and Hoifung Poon. 2018. Deep probabilistic logic: A unifying framework for indirect supervision. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1891–1902, Brussels, Belgium. Association for Computational Linguistics.

Haoyu Wang, Muhao Chen, Hongming Zhang, and Dan Roth. 2020. Joint constrained learning for event-event relation extraction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 696–706, Online. Association for Computational Linguistics.

Jingyi Xu, Zilu Zhang, Tal Friedman, Yitao Liang, and Guy Van den Broeck. 2018. A semantic loss function for deep learning with symbolic knowledge. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 5502–5511. PMLR.

Zhongzhi Xu, Yucan Xu, Florence Cheung, Mabel Cheng, Daniel Lung, Yik Wa Law, Byron Chiang, Qingpeng Zhang, and Paul S.F. Yip. 2021. Detecting suicide risk using knowledge-aware natural language processing and counseling service data. *Social Science & Medicine*, 283:114176.

A Data Anonymization and Storage

The data was anonymized following HIPAA compliance guidelines. We use special mask tokens for identifiable information, including names, locations, ZIP codes, ages, phone numbers, related entities (e.g., school, hospital, etc.), and any other numbers. All the data are stored in a HIPAA-compliant cloud folder. Only staff signed under the IRB approval of this project (`IRB_00131153`) were allowed to have access to the folder. The staff have all been trained with basic knowledge on data confidentiality, privacy, and protection.

B Anonymized examples of sessions

Figures 2 and 3 show example snippets of encounters with associated D-codes with Lower and Higher risk assessment respectively.

👨🏽‍🦱	I have depression. It's not super severe. I have depression in my family history so I'm not surprised. I just want to talk to someone to see if I can make sense of it.	No-D
👩	I am here for you! Can you say more about your feeling depressed?	
👨🏽‍🦱	I feel hopeless most of the time and have little motivation to go to places.	No-D
👩	Sometimes It can be hard to identify our feelings like this. Have you ever had any kind of thought of killing yourself? just want to make sure you are safe.	
👨🏽‍🦱	I did have it a while ago but I'm okay. I'm not suicidal.	D_1
👨🏽‍🦱	And by a while I mean [#DATE] ago.	D_1
👩	I'm glad to hear that you are okay right now. Can you tell me more about your life?	
	⋮	

Figure 2: Example snippet of an anonymized Lower risk session with associated utterance-level labels (D-codes) for client (👨🏽‍🦱) utterances.

👨🏽‍🦱	I think I'm not safe right now.	No-D
👩	Hi, I'm sorry to hear that. What happened?	
👨🏽‍🦱	I'm behind in school and they will not let me graduate. My mom is pissed off.	No-D
👩	Who doesn't let you graduate?	
👨🏽‍🦱	I don't want to be alive anymore. I took some pills.	D_2, D_{2a}, D_{2b}, D_3
👩	What pills did you take?	
👨🏽‍🦱	I did this before but I was stopped. That's all I need. I don't want to be there anymore.	D_1, D_9
	⋮	

Figure 3: Example snippet of an anonymized Higher risk session with associated utterance-level labels (D-codes) for client (👨🏽‍🦱) utterances.

C Reproducibility

C.1 Encoding Model

We pre-process both datasets U and E by prepending special tokens indicating the originator of each utterance in a session: we added the token [#COUNSELOR] or [#CLIENT] to counselor and client utterances accordingly. Each utterance is then encoded with a domain-adapted RoBERTa model of 768-dimensional outputs.

Before the utterance encoding, we add the originator tokens to the matrix embedding of RoBERTa. We respectively initialize these tokens by averaging the pre-trained embeddings of the words "client", "counselor" with corresponding direct synonyms (e.g.,"patient"/"therapist"). Similarly, we add and initialize the special anonymization mask tokens (e.g., [#SCHOOL], [#ZIP-CODE], [#PERSON]). Following Gururangan et al. (2020), we adapt the RoBERTa-base from the hugginface library using 2 million general mental health counseling (crisis,tips,support) unlabeled utterances extracted from SafeUT. We continue training RoBERTa-base for 5 epochs with AdamW optimization, learning rate of $5e-5$, batch size 4, and using a mask language model head with masking probability of 0.15 (seed=1). To obtain the utterance RoBERTa encondings, we average the concatenation of the last four hidden states of the adapted RoBERTa-base outputs from the truncated (max length 512) input utterance tokens. The resulting utterance encodings are 3072-dimensional vectors.

C.2 Architecture

On top of the RoBERTa utterance embeddings, we use two trasformer encoder layers. Each transformer layer has 8 heads, 2048 feedforward dimension, ReLU activation on the intermediate layer and 1e-5 eps stability value at the normalization layer. We applied a positional encoding layer with dropout probability of 0.2 and a eps value of 1e-12 to the input utterance embeddings before the transformer block. In all, our system has 275 million parameters.

C.3 Full System Description

Let $e = \{m_1, m_2, \ldots, m_n\}$ be an input session. For each utterance $m_i \in e$, we denote as m_i^* the corresponding utterance RoBERTa embedding obtained as described in C.1.

$$\text{RoBERTa}(m_i) = m_i^*$$

We input the encoded encounter $e^* = \{m_1^*, m_2^*, \ldots, m_n^*\}$ into the transformer block to obtain a list of session-contextualized utterance embeddings $\{u_1, u_2, \ldots, u_n\}$

$$\text{Transformer}(e^*) = \{u_1, u_2, \ldots, u_n\} = u$$

To obtain an entire session embedding s we average (as described in C.1) the transformer utterance embeddings

$$\text{average}(\{u_1, u_2, \ldots, u_n\}) = s$$

We apply a linear layer P_u of length 9 and an element-wise sigmoid activation to each client utterance $u_c \in u$ obtaining a nine-dimensional vector

$\sigma(\mathrm{P}_u(u_c))$. Each entry in $\sigma(\mathrm{P}_u(u_c))$ represents the probability that the utterance u_c having each of the D-codes is True. For instance, the first and second coordinates of $\sigma(\mathrm{P}_u(u_c))$ are the probabilities that the facts $\texttt{NoCode}(u_c)$ and $\texttt{HasCode}(u_c, \mathrm{D}_1)$ respectively hold. This is,

$$\sigma(\mathrm{P}_u(u_c))[1] = [\texttt{NoCode}(u_c)]$$

and

$$\sigma(\mathrm{P}_u(u_c))[2] = [\texttt{HasCode}(u_c, \mathrm{D}_1)]$$

Similarly, we apply a linear layer P_s of length 2 and a softmax activation to the session embedding s obtaining a two-dimensional vector $\text{softmax}(\mathrm{P}_s(s))$. Here, we have that

$$\text{softmax}(\mathrm{P}_s(s))[1] = \texttt{Risk}(e, \text{Lower})$$

and

$$\text{softmax}(\mathrm{P}_s(s))[2] = \texttt{Risk}(e, \text{Higher})$$

We use the relaxed truth-values in the utterance and encounter vectors—$\sigma(\mathrm{P}_u(u_c))$ and $\text{softmax}(\mathrm{P}_s(s))$—to compute all the loss components in (15) using the R-product logic.

We do not fine-tune the underlying domain adapted RoBERTa model due to hardware limitations. The data for this project is housed in a secure compute infrastructure whose GPUs size do not allow us to load entire input sessions and their gradients in memory.

C.4 Experimental setting

Multiple runs We train the system using the training splits of the utterance U and session E datasets using 5 different seeds (0,1,2,3,4).

Data Batching and optimization We randomly select batches from U (we denote B_U) and E (denoted B_E) until completing each epoch. For B_U batches we have the labels to compute the utterance multi-label loss L_U and not the session binary loss L_E, therefore the latter does not contribute during back-propagation. Similarly, input B_E batches update the L_E loss but not the L_U loss. Importantly, the unsupervised losses L_{Joint}, L_{D} and L_{NoCode} can be computed from both B_U and B_E batches. We use rescaling weights on L_U and L_E to compensate label imbalance. In this setting, the size of a batch is defined by the number of sessions, and sessions can have different sizes in terms of contained utterance. Hence, we normalize the loss for B_U batches (also for B_E batches for implementation convenience) by averaging the utterance losses from all sessions in the batch. This strategy makes the system performance more stable across epochs.

Training with rules The MT+Rules system reported in the tables from section 5.2 is obtained from training the baseline Multi-Task (MT) system for 75 epochs until convergence and then continue training adding the rules for 75 epochs more. We found that this strategy mitigates high variance in performance across different runs.

Evaluation and model selection We run hyper-parameter tuning for 75 epochs, and then train with the best combination for 150 epochs (using seed 1). We select the model from the epoch with best micro averaged F_1 over the client utterances labeled with at least one D-code in the development split of the set U. We stop training after 50 epochs of non-increase in F_1 and keep the model from the latest best epoch.

Hyper-parameter tuning details The hyper-parameter search space is the following:

- Learning rate (lr): $1e$-4, $2e$-4, $5e$-4, $1e$-5, $2e$-5, $5e$-5, $1e$-6, $2e$-6, $5e$-6
- λ's (eq. 15) : 0.0001, 0.001, 0.01, 0.1, 1, 5, 10
- Batch size (bs): 4, 8, 16

Due to the size of the search space we do not perform full grid hyper-parameter search for all the systems reported. We first select the best hyper-parameters exploring the search space for the baseline models that only includes learning rate, batch size, and λ_E (for the multi-task baseline). From this process we discover values for which the baselines do not converge, and discard them for the subsequent search—when adding the rules into the system. For instance, we discard the learning rate values $1e-4$, $2e-4$, $5e-4$, $1e-6$, $2e-6$, $5e-6$, and the λ_E values $0.1, 1, 5, 10$. We further reduce the search space by incrementally adding rules into the system and exploring the influence of different λ values. For instance, we observe that the multi-task baseline system trained using only the NoCode rule under-performs with λ_{NoCode} values smaller below 1. Due to the running time of each hyper-parameter combination, this aggressive pruning strategy was necessary to make the

	lr	bs	λ_E	λ_{NoCode}	λ_{D}	λ_{Joint}
Baseline	$2e$-4	16	-	-	-	-
MT	$5e$-5	4	$1e$-3	-	-	-
MT+Rules	$2e$-5	8	$1e$-4	1	$1e$-4	$1e$-3
MT+NoCode	$2e$-5	4	$1e$-3	10	-	-
MT+D-rules	$5e$-5	16	$1e$-4	-	$1e-3$	-
MT+Joint	$5e$-5	4	$1e$-4	-	-	0.01

Table 10: Best hyper-parameter combinations used to train the models reported in section 5.2

experiments feasible. Table 10 shows the hyper-parameter combinations used to train the models reported in section 5.2.

Running times We give an approximate estimated time for each stage of our experiments with significant running time:

1. Adapting RoBERTa with `SafeUT` utterances: 4 days
2. RoBERTa session encodings: 4 hours for the risk assessment set E, and 30 minutes for utterance risk status set U.
3. One training epoch: 1.2 hours for each hyper-parameter combination including evaluation for development and rules violations.

Code and computing infrastructure We implemented all our experiments in Python, using the PyTorch, Pandas, and scikit-learn libraries. We used a server located in an IRB approved HIPAA protected environment with the following configuration:

- CPU: Intel (R) Xeon (R), E5-2640, 2.40 GHz
- GPU: NVIDIA TITAN X (Pascal)
- RAM 12GB

D Logic Relaxations

D.1 Implementation details

As discussed in Xu et al. (2018) t-norm logic relaxations are syntactic, rather than semantic, representations of boolean statements. In our particular case, the relaxation of a predicate rule may produce a different loss than the relaxation of its contrapositive. From this perspective, to obtain signal from syntactically different but semantically identical representations of the constraints in our system, we also add their respective contrapositive in the learning loss.

	$\mathcal{S}$-Gödel	$\mathcal{R}$-Product
$\wedge$	$\min(a, b)$	$a \cdot b$
$\neg$	-	$1 - a$
$\vee$	$\max(a, b)$	$a + b - a \cdot b$
$\rightarrow$	-	$\min(1, \frac{b}{a})$

Table 11: Relaxation definition for the basic logical connectives as presented in Medina Grespan et al. (2021). The letters a and b denote the relaxed truth values of the arguments of the formulas. In the implication definitions, a and b denote the antecedent and the consequent respectively.

In the declarative definition of the loss, we can incorporate each constraint along its contrapositive in two logically equivalent ways – as a conjunction or as a disjunction. For instance, let F_1 and F_2 be Boolean formulas. A constraint of the form $F_1 \rightarrow F_2$, can be added along its contrapositive into a declarative boolean statement as the conjunctive term $(F_1 \rightarrow F_2) \wedge (\neg F_2 \rightarrow \neg F_1)$ or as the disjunctive term $(F_1 \rightarrow F_2) \vee (\neg F_2 \rightarrow \neg F_1)$. Although the latter equivalent expressions also generate different relaxation signals, we found through preliminary experiments that adding the constraint-contrapositive disjunction terms accelerates system convergence.

As an example, by adding the contrapositive to the joint constraint (5) we obtain:

$$\forall e \in E \cup U,\ \forall m \in e,\ \forall d \in \mathcal{H}$$
$$\big(\texttt{Risk}(e, \text{Lower}) \rightarrow \neg\texttt{HasCode}(m, d)\big) \vee$$
$$\big(\texttt{HasCode}(m, d) \rightarrow \texttt{Risk}(e, \text{Higher})\big) \quad (16)$$

We use the $\mathcal{S}$-Gödel over the $\mathcal{R}$-Product logic (Table 11) to relax the disjunction of the rule and its contrapositive. We encounter that taking the maximum of the disjuncts, as defined by $\mathcal{S}$-Gödel,

provides better learning stability (the maximum function becomes the minimum after taking the negative logarithm for optimization). In this way, we implement the relaxation of the joint rule from equation (16) as:

$$\sum_{e \in U \cup E} \sum_{m \in e} \sum_{d \in \mathcal{H}} \min \left(\ell_{\text{J}} \left(e, m, d \right), \ell_{\text{C}} \left(e, m, d \right) \right) \tag{17}$$

where, $\ell_{\text{J}} \left(e, m, d \right)$ is defined in equation (11), and

$$\ell_{\text{C}} \left(e, m, d \right) = \text{ReLU}\big(\log([\texttt{HasCode}(m, d)]) - \log[\texttt{Risk}(e, \text{Higher})] \big) \tag{18}$$

The D-Code and NoCode constraints are implemented following an analogous strategy.

E Error Analysis

Table 12 shows examples of the four types of errors discussed in the paper.

F D-code rules

Table 13 shows the full set of dependencies between the D-codes.

Error type	Total %	Examples		
		Gold	Predicted	Utterance
Passive vs. Ideation	27%	D_{2b}	D_{2b}, D_2	I am having those thoughts again. Being better off dead.
		D_2	D_{2b}	I need a reason not to kill myself. I've let myself become the very thing I hate.
Lifetime vs. Current	15%	D_2	D_1	I'm worried. She has sent me a text saying she was going to commit suicide.
		D_1	D_2	One day I just woke up with the feeling.
		D_1	D_1, D_2	He told us last night that he is suicidal. What should we do?
No Code	20%	D_6	No-D	Not really.
		D_2, D_{2a}	No-D	Yes...
		D_6, D_{6b}	No-D	But I know where the key is.
Commonsense Knowledge	4%	D_2	No-D	A kid on social media posted bloody cuts, the caption said bye bye!...
		D_{2b}	No-D	I know I am only alive for my friends and food! LOL
		D_1, D_6	D_1	Earlier I was wading through a fast moving creek wanting to lie down in the rapid part.

Table 12: Error analysis of the MT+Rules model on the development split of the utterance risk status dataset U. The first column lists the four main types of errors we found, the second column indicates the percentage of the total errors (false positives and false negative) corresponding to the type of mistake, the third column shows representative examples of miss-classified client utterances (modified from the real data for anonymity)—we respectively report the gold and predicted D-codes for each utterance.

Prior attempts (D_9)	implies	Lifetime ideation (D_1)
Imminent risk (D_{2a})	implies	Current ideation (D_2)
Attempt in progress (D_3)	implies	Current ideation (D_2)
Attempt in progress (D_3)	implies	Imminent risk (D_{2a})
Attempt in progress (D_3)	implies	Method chosen or considered (D_6)
Attempt in progress (D_3)	implies	Availability of means (D_{6b})
Availability of means (D_{6b})	implies	Method chosen or considered (D_6)

Table 13: List of existing dependencies between D-codes. We denote this list of logical constraints as RULES.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
9

☑ A2. Did you discuss any potential risks of your work?
9,10

☑ A3. Do the abstract and introduction summarize the paper's main claims?
1

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☒ Did you use or create scientific artifacts?

Left blank.

☐ B1. Did you cite the creators of artifacts you used?
No response.

☐ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
No response.

☐ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
No response.

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
No response.

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
No response.

☐ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
No response.

C ☑ Did you run computational experiments?

5

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
C

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
5,C

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
5,C

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
4, C

D ☑ Did you use human annotators (e.g., crowdworkers) or research with human participants?

We work with data about humans, but do not research with human subjects

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
Not applicable. We used annotated data. We did not annotate data in this project.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
Not applicable. Left blank.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
Not applicable. Left blank.

☑ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
A

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
Not applicable. Left blank.