CredenceAI at SemEval-2026 Task 10: A Span-Consistency Network with Cross-Marker Attention for Conspiracy Marker Extraction

Ishaan Karan
International Institute of Information Technology Hyderabad
ishaan.karan@research.iiit.ac.in

Abstract

We describe our system for SemEval-2026 Task 10, Subtask 1: Conspiracy Marker Extraction (Samory et al., 2026), which involves identifying spans of five marker types (Actor, Action, Effect, Evidence, and Victim) in English social media text. Standard token-level classifiers predict each position independently, which can cause span fragmentation and ignores inter-marker dependencies. We present a Span-Consistency Network (SCN) built on DeBERTa-v3-large designed to address these issues. First, a Span Consistency Layer (SCL) propagates span-level signals to constituent tokens via a min-over-span, max-over-position mechanism. Second, Cross-Marker Attention (CMA) models co-occurrence patterns between marker types through a learned correlation matrix. Third, we adopt a recall-biased Tversky loss augmented with Span Count Regularization (SCR), which penalizes mismatches between predicted and gold span counts, penalizing diffuse probability distributions in favor of discrete spans. We ensemble five models trained via stratified cross-validation with probability averaging. Our system achieved a Macro F1 of 0.24 on the test set, placing second among participating teams. Ablation experiments on the development set show that SCR has a large effect: its removal leads to prediction collapse for three of five marker types, and that the Tversky loss outperforms binary cross-entropy. We additionally provide a latency analysis of CMA, an analysis of how SCL handles cross-type overlapping spans, and an ablation on SCR warm-up scheduling.

1 Introduction

Conspiracy theories have become a pervasive feature of online discourse, influencing public opinion on topics ranging from public health to electoral integrity. Understanding the rhetorical structure of conspiracy narratives is relevant to content moderation, media literacy, and social science research (Da San Martino et al., 2019). SemEval-2026 Task 10 (Samory et al., 2026) operationalizes this as a structured span extraction task over English social media text: given a document, systems must identify character-level spans corresponding to five marker types grounded in evolutionary psychology: *Actor* (who is responsible), *Action* (what they are doing), *Effect* (the consequences), *Evidence* (the cited justification), and *Victim* (who is harmed). The task is evaluated using overlap-based Macro F1, where predicted and gold spans are matched via greedy IoU alignment, and performance is averaged equally across all five types.

This task presents three challenges that standard token classification approaches handle poorly. First, token-level classifiers predict each position independently, which produces fragmented spans when adjacent tokens receive inconsistent scores; this is particularly acute for long markers like Evidence and Effect that can span entire clauses. Second, the five marker types tend to co-occur: Actor mentions typically accompany Actions, and Evidence spans tend to appear alongside the Actions they justify. Independent per-marker prediction ignores these dependencies. Third, the class distribution is heavily skewed toward the negative class (most tokens are not part of any marker), causing standard losses like binary cross-entropy to favor conservative, low-recall predictions.

We present a Span-Consistency Network (SCN) designed to address these challenges. A Span Consistency Layer (SCL) encourages coherent span boundaries by propagating span-level confidence signals back to individual tokens. Cross-Marker Attention (CMA) models inter-marker correlations through a learned dependency matrix. A recall-biased Tversky loss with Span Count Regularization (SCR) directly penalizes span fragmentation and penalizes the model for distributing probability mass diffusely across tokens.

Our system achieved a Macro F1 of 0.24 on the

Proceedings of the The 20th International Workshop on Semantic Evaluation (2026), pages 3240–3249

held-out test set, placing second among participating teams. Ablation experiments on the 100-sample development set reveal that SCR is the most impactful component: removing it causes three of five marker types to receive zero predictions entirely, as the model's sigmoid outputs never cross the classification threshold. The Tversky loss provides consistent gains over binary cross-entropy, and the full system outperforms a DeBERTa+MLP baseline by 8 points Macro F1. This paper additionally contributes: (i) a latency benchmark isolating the cost of CMA from that of SCL; (ii) an analysis of how SCN handles overlapping spans, both mechanistically and empirically; (iii) an SCR warm-up ablation showing that delayed activation *harms* performance; (iv) a CRF baseline comparison; and (v) a discussion of generalizability concerns around manually configured priors.

2 Background

2.1 Task Setup and Evaluation

SemEval-2026 Task 10 targets conspiracy marker extraction from English-language social media posts (Samory et al., 2026). Each document may contain zero or more spans for each of five marker types: Actor (individual or group agents), Action (what the actor is doing), Effect (consequences), Victim (individuals or groups harmed), and Evidence (claims supporting the theory). Spans are defined by character-level offsets. Notably, spans of different types may overlap (the same fragment can be both Actor and Victim), requiring multi-label predictions at each position.

The training set consists of 4,316 annotated posts containing 21,965 marker annotations. Marker frequencies are imbalanced: Actor is most frequent (6,416) while Victim is least (3,315), nearly 2:1. Span lengths vary substantially: Actor and Victim spans are short (mean $\sim$13 characters, typically noun phrases), while Effect and Evidence spans are long (mean $\sim$35 characters, often full clauses). The development set contains 100 samples. Gold labels for the test set were not released at the time of writing.

The official metric is overlap-based Macro F1. For each marker type, predicted and gold spans are matched greedily by descending IoU. A matched prediction is a true positive; unmatched predictions are false positives and unmatched gold spans false negatives. Per-type F1 scores are averaged equally to produce Macro F1. This metric rewards partial overlap but penalizes both boundary imprecision and missed spans.

2.2 Related Work

Span extraction has been widely studied in named entity recognition (Devlin et al., 2019; Lample et al., 2016), with recent work exploring span enumeration (Eberts and Ulges, 2020), biaffine scoring (Zhong and Chen, 2021; Yu et al., 2020), and unified sequence-to-sequence formulations (Raffel et al., 2023). For class-imbalanced sequence labeling, Li et al. (2020) demonstrated Dice loss as a differentiable F1 surrogate. The Tversky loss (Salehi et al., 2017; Tversky, 1977) generalizes Dice with asymmetric false-positive and false-negative weighting, originally for medical image segmentation. Generalized Dice loss (Sudre et al., 2017) addresses class imbalance through per-class weighting. Our Span Count Regularization extends these set-based losses with a structural constraint on the number of predicted spans. Cross-marker interaction has been explored in joint entity and relation extraction, though typically through shared encoders rather than explicit correlation modeling.

3 System Overview

Our architecture follows a token classification paradigm with three extensions for multi-label span extraction. The pipeline is illustrated in Figure 1 and proceeds as: (1) a pretrained encoder produces contextualized representations, (2) per-marker scoring heads produce token-level logits, (3) a Span Consistency Layer propagates span-level signals, (4) Cross-Marker Attention refines predictions via inter-marker dependencies, and (5) thresholded decoding produces character-level spans.

3.1 Encoder and Token Scoring

We use DeBERTa-v3-large (He et al., 2023) as the backbone encoder, producing contextualized representations $\mathbf{h}_i \in \mathbb{R}^d$ (d=1024). DeBERTa's disentangled attention, which separately encodes content and position, has been effective across NLP tasks.

For each marker type $m \in \mathcal{M}$, a dedicated scoring head maps encoder output to a scalar logit: $r_i^{(m)} = \text{MLP}_m(\mathbf{h}_i) \in \mathbb{R}$. Each MLP_m has two hidden layers (512, 256) with GELU activations and dropout (p=0.1). Separate heads allow each marker type to learn different boundary patterns.

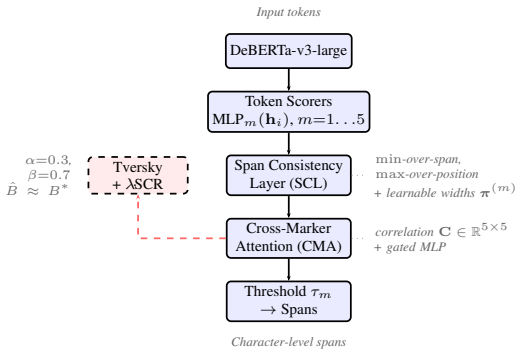

Figure 1: SCN architecture. Dashed box shows the training loss. SCL propagates min–max span signals to tokens. CMA refines predictions via inter-marker correlations.

3.2 Span Consistency Layer

A known limitation of token-level classification (Lample et al., 2016) is that predictions at adjacent positions are made independently, producing fragmented spans. The Span Consistency Layer (SCL) addresses this by computing span-level confidence signals and propagating them back to constituent tokens.

For each marker m with maximum span width prior W_m (set from data statistics; e.g., W_{Actor}=6, W_{Evidence}=14), SCL computes the minimum token score within each window of width $w \in \{2, \ldots, W_m\}$:

$$c^{(m)}_{a,a+w-1} = \min_{k \in [a,\, a+w-1]} r^{(m)}_k \tag{1}$$

This *weakest-link* indicates span coherence: if all tokens score high, the minimum is high. Span-level signals are weighted by learnable width preferences $\boldsymbol{\pi}^{(m)} = \text{softmax}(\boldsymbol{\ell}^{(m)}) \in \mathbb{R}^{W_m}$, allowing the model to learn that certain widths are more probable per marker. These are scattered back to tokens by taking the maximum over all covering spans:

$$\text{boost}^{(m)}_i = \max_{\substack{(a,b):a\le i\le b \\ 2\le b-a+1\le W_m}} \pi^{(m)}_{b-a+1} \cdot c^{(m)}_{a,b} \tag{2}$$

The boosted score combines original and span-level signals via a learnable gate γ_m:

$$\hat{r}^{(m)}_i = r^{(m)}_i + \gamma_m \cdot \text{boost}^{(m)}_i \tag{3}$$

An important property is that $\hat{r}^{(m)}_i$ depends on scores of tokens *other than* i through min and max. If surrounding tokens all score highly (forming a plausible span), the boost is high, encouraging span inclusion; isolated tokens receive no boost. This is intended to reduce span fragmentation.

Independence across marker types. The SCL operates *independently per marker type*: each m has its own W_m, its own width logits $\boldsymbol{\pi}^{(m)}$, its own gate γ_m, and its own min/max computation over the type-specific logits $r^{(m)}_{\cdot}$. Crucially, the min in Equation (3) never crosses marker types, so a position identified as belonging to a span of type m exerts no suppressive influence on type m'. This is what enables overlapping spans of different types to share the same character offsets, a property we analyze empirically in Section 5.4.

3.3 Cross-Marker Attention

Conspiracy markers tend to co-occur: Actor typically appears with Action, and Evidence accompanies the Action it supports. Cross-Marker Attention (CMA) models these dependencies by allowing predictions across types to influence each other.

Given stacked boosted logits $\hat{\mathbf{R}} \in \mathbb{R}^{N\times M}$ (M=5), CMA computes per-token marker probabilities, applies a learnable correlation matrix $\mathbf{C} \in \mathbb{R}^{M\times M}$ (initialized to $0.5\,\mathbf{I}$), and passes the result through a gated MLP:

$$\mathbf{R}' = \hat{\mathbf{R}} + \sigma(g) \cdot \text{MLP}_{\text{cross}}\Big(\sigma(\hat{\mathbf{R}}) \cdot \mathbf{C}\Big) \tag{4}$$

where g is a learnable gate (initialized to 0.3) and σ denotes sigmoid. The MLP projects to a 64-dimensional hidden space with GELU and back to M dimensions. $\mathbf{C}$ learns, e.g., that high Actor probability should boost Action at the same position. Importantly, CMA is a *per-token* operation that mixes type predictions but not spatial context, so it preserves the per-type independence of SCL while injecting type-coupling signals.

3.4 Loss Function

Tversky Loss. Standard BCE treats each token equally, which on imbalanced data favors the majority negative class. Following Salehi et al. (2017), we adopt asymmetric parameters α=0.3, β=0.7:

$$\mathcal{L}^{(m)}_T = 1 - \frac{\text{TP}_m + \epsilon}{\text{TP}_m + \alpha \cdot \text{FP}_m + \beta \cdot \text{FN}_m + \epsilon} \tag{5}$$

where TP, FP, FN are soft counts from sigmoid probabilities $p^{(m)}_i = \sigma(R'_{i,m})$. The asymmetry ($\beta$>$\alpha$) penalizes false negatives more heavily, encouraging broader coverage.

Span Count Regularization. Tversky loss on soft probabilities can be minimized by distributing activation diffusely (e.g., 0.3 on 20 tokens) rather than forming discrete spans (0.9 on 3 tokens). Both yield similar soft TP values, but only the latter survives thresholding. We formalize this below; further analysis (SCR as total variation constraint, consistency of $\hat{B}$) is in Appendix B.

Proposition 1 (Tversky admits diffuse minimizers)**.** *Let $\mathbf{y} \in \{0,1\}^N$ have K positive tokens ($K \ll N$). The uniform prediction $p_i = c$ for all i yields Tversky loss:*

$$\mathcal{L}_T(c) = 1 - \frac{cK + \epsilon}{cK + \alpha c(N-K) + \beta(1-c)K + \epsilon} \tag{6}$$

Setting $\partial \mathcal{L}_T / \partial c = 0$ gives a minimizer $c^* \in (0,1)$. For typical span detection where $N-K \gg K$, c^* is small (e.g., 0.2–0.4). Under threshold decoding with $\tau > c^*$, this produces zero predicted spans despite low loss.

SCR prevents this by constraining the total variation of the prediction sequence. The predicted span count is:

$$\hat{B}^{(m)} = \frac{1}{2} \sum_{i=1}^{N-1} \left| p_{i+1}^{(m)} - p_i^{(m)} \right| = \frac{1}{2} \, \mathrm{TV}(\mathbf{p}^{(m)}) \tag{7}$$

The diffuse solution has TV $= 0$, so $\mathcal{L}_{\text{SCR}} = (B^*)^2 > 0$ whenever gold spans exist, penalizing it. For binary sequences, $\hat{B}$ equals the exact span count; for soft probabilities, it is a consistent underestimate (Appendix B), so minimizing SCR pushes $\mathbf{p}$ toward sharper transitions. With gold count $B^{*(m)}$ from label transitions, the SCR penalty is $\mathcal{L}_{\text{SCR}}^{(m)} = (\hat{B}^{(m)} - B^{*(m)})^2$. The total loss:

$$\mathcal{L} = \frac{\sum_m c_m \left[\mathcal{L}_T^{(m)} + \lambda \cdot \mathcal{L}_{\text{SCR}}^{(m)} \right]}{\sum_m c_m} \tag{8}$$

where λ=0.15 and per-marker weights c_m upweight rarer types (c_{Victim}=1.20, c_{Actor}=0.85) for Macro F1 optimization.

3.5 Ensemble and Decoding

We train five models via stratified 5-fold cross-validation (Breiman, 1996), where stratification uses a binary encoding of marker type presence. At inference, sigmoid probabilities are averaged element-wise across all five models. Per-marker classification thresholds are tuned on the development set via grid search using the full model, then held fixed across all ablation configurations. Consecutive above-threshold tokens are grouped into spans and converted to character offsets via the tokenizer's offset mapping.

4 Experimental Setup

Training. Each fold model trains on ∼3,453 samples (80%) for 12 epochs. The held-out 20% is not used for early stopping or model selection. We use AdamW (Loshchilov and Hutter, 2019) with differential learning rates: 2×10^{-5} for the pretrained encoder and 1×10^{-4} for task-specific parameters. Effective batch size is 16 (micro-batch 2, accumulation 8). Cosine LR schedule with 10% linear warmup. FP16 mixed precision. Max sequence length is 256 tokens.

Evaluation. All ablation results use the official overlap-based Macro F1 on the 100-sample dev set. Per-marker thresholds are tuned via grid search on the full model and held fixed across all configurations, so differences reflect model quality, not threshold optimization.

Software. PyTorch 2.0, HuggingFace Transformers (Wolf et al., 2020). Training takes ∼2 hours per fold on a single NVIDIA RTX 2080 Ti (11 GB). Full hyperparameters in Appendix A.

5 Results

5.1 Official Results

Our system achieved a Macro F1 of 0.24 on the held-out test set, placing second among participating teams. On the development set, the full system achieves 0.40 Macro F1. The gap (0.16 points) likely reflects distribution differences between the splits and the small dev set size (100 samples); we discuss this further in Appendix C.

5.2 Ablation Study

We train variants that each remove one component while keeping others fixed. All use the same hyperparameters. Thresholds are tuned on the full model and held fixed for all configurations. Results are in Tables 1 and 2.

SCR has a large effect. Removing Span Count Regularization causes a large drop from 0.40 to 0.17 Macro F1. Three markers (Effect, Evidence, and Victim) receive exactly zero predictions: sigmoid probabilities never reach the tuned thresholds. Table 3 is consistent with this explanation: without

Configuration	F1	P	R
Full SCN	**0.40**	0.39	**0.44**
w/o SCR (λ=0)	0.17	0.18	0.17
w/o Tversky (BCE)	0.34	**0.41**	0.29
DeBERTa+CRF (BIO)	0.34	–	–
DeBERTa+MLP+BCE (baseline)	0.32	0.36	0.28

Table 1: Ablation results on the 100-sample dev set (Macro F1). All configurations use thresholds tuned on the full model. Baseline uses the same encoder with independent MLP heads and BCE. The CRF baseline uses one linear-chain CRF per marker type over BIO tags.

	Act.	Actr.	Eff.	Evi.	Vic.
Full	.34	**.55**	**.34**	**.32**	**.46**
w/o SCR	**.36**	.51	.00	.00	.00
w/o Tvk	.32	.52	.29	.17	.39
CRF	.32	.51	.23	.23	.43
Baseline	.30	.46	.21	.19	.42

Table 2: Per-marker F1 on dev. Without SCR, Effect, Evidence, and Victim collapse to zero predictions. The CRF baseline matches the non-SCN models on short markers (Actor, Victim) but trails the full SCN on long-span markers (Effect, Evidence) by 9–11 points.

SCR, the soft boundary count $\hat{B}^{(m)}$ collapses to near zero for these markers (Effect: 0.08, Evidence: 0.20, Victim: 0.00), meaning the model forms no span-like probability transitions at all. With SCR, $\hat{B}$ rises to 0.86, 0.71, and 0.65 respectively, matching the expected span structure. The Tversky loss alone can be minimized by distributing activation diffusely rather than forming discrete spans; SCR encourages sharper predictions by penalizing this mismatch.

Effect of Tversky loss on recall. Replacing the recall-biased Tversky loss (α=0.3, β=0.7) with BCE reduces F1 from 0.40 to 0.34, accompanied by recall dropping from 0.44 to 0.29. Evidence suffers the largest per-type drop ($0.32 \rightarrow 0.17$), as this marker is both rare and spans long fragments that BCE-trained models tend to under-predict.

Comparison with structured baselines. A DeBERTa-v3-large + per-marker linear-chain CRF over BIO tags reaches only 0.34 Macro F1 (Table 1), tying with the BCE ablation and trailing the full SCN by 6 points; the gap is concentrated on long-span markers (Effect, Evidence) where SCL's soft span-level smoothing helps most.

	Act.	Actr.	Eff.	Evi.	Vic.
Full SCN	1.17	1.40	0.86	0.71	0.65
w/o SCR	1.03	1.44	0.08	0.20	0.00

Table 3: Mean soft boundary count $\hat{B}^{(m)}$ per marker on dev. Without SCR, Effect/Evidence/Victim form no span boundaries.

Configuration	Latency (ms/sample)
Full SCN (with CMA)	32.09 ± 0.33
SCN − CMA	32.46 ± 1.40
DeBERTa+MLP (no SCL, CMA)	24.45 ± 1.23
Δ from CMA	-0.38 ms (-1.17%)
Δ from SCL+CMA	$+7.63$ ms ($+31.2\%$)

Table 4: Inference latency, batch size 1, RTX 2080 Ti. Mean $\pm$ standard deviation across the dev set after five warm-up runs.

Full system vs. baseline. The complete SCN outperforms DeBERTa+MLP+BCE by 8 points (0.40 vs. 0.32) across all five types. Ablation of SCL, CMA, and class weights yielded differences within the noise margin on 100 samples and are omitted. The 5-fold ensemble improved performance on the held-out test set.

5.3 Computational Complexity of CMA

A reviewer asked whether the cross-marker attention introduces significant inference overhead relative to a standard transformer head. We measured per-sample latency on the 100-sample dev set, batch size 1, on a single NVIDIA RTX 2080 Ti, averaged over the dev set with five warm-up runs (Table 4). The full SCN runs at 32.09 ± 0.33 ms/sample. Removing CMA changes this only marginally to 32.46 ± 1.40 ms/sample; CMA itself accounts for -0.38 ms (-1.17% of the model without it), within measurement noise. By contrast, removing both SCL and CMA (the DeBERTa+MLP baseline) reduces latency to 24.45 ± 1.23 ms, a 7.63 ms reduction (31.2% of baseline).

The interpretation is that CMA is essentially free at inference: it performs a single $M \times M$ matrix multiplication (with M=5) and a small two-layer MLP over per-token marker probabilities, both of which are negligible relative to the encoder forward pass. The non-trivial overhead comes from SCL, where the per-marker $\min$-over-windows of widths 2 to W_m requires $\sum_m W_m$ unfold operations. Optimizing SCL via a single fused kernel is a natural direction for future work, but the current implementation already runs end-to-end at roughly

30 samples/sec on commodity hardware.

5.4 Handling Overlapping Spans

Reviewers asked how the min-over-span mechanism handles instances where different marker types share the same character offsets. We address this in two parts: (i) the architectural mechanism, and (ii) an empirical analysis on dev data.

Mechanism. As noted in Section 3.2, the SCL is applied independently per marker type. Each marker m has its own scoring head, its own width logits $\boldsymbol{\pi}^{(m)}$, its own gate γ_m, and its own W_m, and the min in Equation (3) only ranges over scores $r_{\cdot}^{(m)}$ of the same type m. As a result, when characters $[a, b]$ are simultaneously a Victim span and an Effect span, the two SCLs run on two disjoint logit streams, each of which can independently produce a high boosted score at every position in $[a, b]$. The CMA layer then mixes the per-type probabilities through the learned correlation $\mathbf{C}$, which can encourage co-activation when types are jointly likely (e.g., the Effect-Victim pairing). Crucially, CMA does this at the same token position without smoothing across positions, so it does not reintroduce fragmentation. The final decoder applies thresholding per marker type independently, so two co-located spans of different types both pass through.

Empirical analysis. On dev, 30 cross-type overlap pairs involve 55 of 456 gold spans (12.1%); on train, 1,934 pairs. The most common pairs (in train, descending) are Effect+Victim, Action+Effect, Action+Victim, Action+Evidence, and Actor+Evidence (each >200 instances). On dev, the dominant pairs are Action+Victim (10) and Effect+Victim (6).

We then asked whether the model handles overlapping spans *better* or *worse* than non-overlapping spans. Restricting to dev gold spans matched by the ensemble: among the 55 gold spans that overlap some other-type gold span, recall is 0.636 (35 detected); among the 401 non-overlapping gold spans, recall is 0.434 (174 detected). Overlapping spans are therefore detected $\sim$1.5$\times$ more reliably than non-overlapping ones. This is consistent with the CMA story: when two marker types are jointly active over the same span, the learned correlation $\mathbf{C}$ provides mutually reinforcing evidence, raising both scores above their thresholds.

SCR schedule (single fold)	Best dev Macro F1
SCR on from epoch 1	**0.564** (epoch 5)
SCR on from epoch 3	0.230 (epoch 1, before SCR)
SCR on from epoch 6	0.255 (epoch 11)

Table 5: Effect of delayed SCR activation, single fold. Numbers are single-fold (not ensemble) and so are not directly comparable to the 0.40 ensemble dev F1. Delaying SCR strongly harms training.

5.5 Does SCR Need Warm-up?

A reviewer asked whether SCR requires a warm-up phase to prevent the model from collapsing to a zero-span trivial solution early in training. This concern follows naturally from Proposition 3 (Appendix B): since $\hat{B}$ is a consistent underestimate of the true span count, an early model that has not yet learned to localize spans could in principle minimize SCR by maximally suppressing all positives.

We tested this directly. Fixing all other hyperparameters and using a single fold, we trained three variants in which SCR is enabled starting from epoch 1, epoch 3, or epoch 6, and compare best dev Macro F1 (Table 5). The result is the opposite of what one might expect: SCR-from-epoch-1 reaches 0.564, while delaying SCR to epoch 3 or 6 collapses dev F1 to 0.230 and 0.255 respectively, with Effect and Victim never recovering above 0.05 once SCR is activated.

The mechanism is the diffuse-minimizer dynamic of Proposition 1. When SCR is off, Tversky training quickly drives the model toward the diffuse fixed point c^*, where the loss landscape is locally flat in the direction of sharpening. Once there, the gradient signal for forming discrete transitions is small, and turning on SCR is insufficient to escape: the model remains stuck in a low-variance regime. Enabling SCR from step 1 prevents the diffuse minimizer from being learned in the first place, so the model never has to climb out of it. We therefore recommend SCR-from-epoch-1 as the default schedule and do not employ warm-up in our final system.

A per-type error analysis (Appendix E) shows that Action spans are over-predicted with large end-boundary errors, that Actor and Victim achieve tight boundaries, and that Evidence and Effect are the hardest types. The dev type confusion matrix (Appendix D, Table 10) reveals an asymmetric Action–Effect coupling: 37 of 71 gold Effects are matched by predicted Actions, vs. 6 in the reverse direction.

Ethical Considerations

Automated conspiracy marker extraction has potential dual-use risks: it can support content moderation and media literacy research, but could also be misused to target individuals or generate more persuasive conspiratorial content. We release our system for research purposes. The training data consists of publicly available social media posts collected and annotated by the task organizers.

Limitations

Our ablation analysis is limited by the 100-sample development set; small differences between configurations cannot be distinguished from noise. The gap between dev (0.40) and test performance (0.24) suggests distribution differences between splits; we observe this is consistent with a different span length distribution and topical mix on dev (Appendix C).

Generalizability of manual priors. Our system relies on two manually configured sets of values: the per-marker maximum span widths W_m (e.g., 6 for Actor, 14 for Evidence) and the per-marker class weights c_m (e.g., 1.20 for Victim, 0.85 for Actor). We selected both from training-set statistics for this dataset specifically, and they are unlikely to transfer to span extraction tasks with different marker inventories or different length distributions. A natural next step would be to learn W_m as part of the model: one option is to parameterize width preferences continuously over a maximum bound (e.g., $W_{\max}$=30) and rely on the softmax over learnable width logits $\boldsymbol{\pi}^{(m)}$ already present in SCL to drive unhelpful widths to zero, allowing the effective W_m to emerge from training. For class weights, one could replace fixed c_m with a gradient-based macro-F1 surrogate or a learned uncertainty-weighted combination (Kendall et al., 2018). We did not pursue these extensions because of compute limitations (each variant requires a full five-fold retrain), but flag them as the most direct route to making the architecture transfer to other span extraction tasks.

Subtask 2 not addressed. Our system targets Subtask 1 (Marker Extraction) only. We did not adapt the architecture for the Subtask 2 detection challenge.

Acknowledgments

We thank the SemEval-2026 Task 10 organizers for providing the dataset and evaluation infrastructure, and the anonymous reviewers for suggestions that motivated several of the analyses in this paper.

References

Leo Breiman. 1996. Bagging predictors. *Machine Learning*, 24(2):123–140.

Giovanni Da San Martino, Alberto Barrón-Cedeño, and Preslav Nakov. 2019. Findings of the NLP4IF-2019 shared task on fine-grained propaganda detection. In *Proceedings of the Second Workshop on Natural Language Processing for Internet Freedom: Censorship, Disinformation, and Propaganda*, pages 162–170, Hong Kong, China. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Markus Eberts and Adrian Ulges. 2020. Span-based joint entity and relation extraction with transformer pre-training. In *ECAI 2020*. IOS Press.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2023. DeBERTaV3: Improving DeBERTa using ELECTRA-style pre-training with gradient-disentangled embedding sharing. *Preprint*, arXiv:2111.09543.

Alex Kendall, Yarin Gal, and Roberto Cipolla. 2018. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. *Preprint*, arXiv:1705.07115.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. *Preprint*, arXiv:1603.01360.

Xiaoya Li, Xiaofei Sun, Yuxian Meng, Junjun Liang, Fei Wu, and Jiwei Li. 2020. Dice loss for data-imbalanced NLP tasks. *Preprint*, arXiv:1911.02855.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. *Preprint*, arXiv:1711.05101.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2023. Exploring the limits of transfer learning with a unified text-to-text transformer. *Preprint*, arXiv:1910.10683.

Seyed Sadegh Mohseni Salehi, Deniz Erdogmus, and Ali Gholipour. 2017. Tversky loss function for image segmentation using 3D fully convolutional deep networks. *Preprint*, arXiv:1706.05721.

Mattia Samory, Felix Soldner, and Veronika Batzdorfer. 2026. SemEval-2026 task 10: PsyCoMark – psycholinguistic conspiracy marker extraction and detection. In *Proceedings of the 20th International Workshop on Semantic Evaluation (SemEval-2026)*, San Diego, United States. Association for Computational Linguistics.

Carole H. Sudre, Wenqi Li, Tom Vercauteren, Sebastien Ourselin, and M. Jorge Cardoso. 2017. Generalised Dice overlap as a deep learning loss function for highly unbalanced segmentations. In *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*, pages 240–248. Springer.

Amos Tversky. 1977. Features of similarity. *Psychological Review*, 84(4):327–352.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, and 3 others. 2020. HuggingFace's transformers: State-of-the-art natural language processing. *Preprint*, arXiv:1910.03771.

Juntao Yu, Bernd Bohnet, and Massimo Poesio. 2020. Named entity recognition as dependency parsing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6470–6476, Online. Association for Computational Linguistics.

Zexuan Zhong and Danqi Chen. 2021. A frustratingly easy approach for entity and relation extraction. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 50–61, Online. Association for Computational Linguistics.

A Hyperparameters and Data Statistics

Tables 6 and 7 list all hyperparameters and per-marker training statistics.

Parameter	Value
Encoder	DeBERTa-v3-large
Max length	256
Batch size (effective)	16 (2×8)
LR (encoder / heads)	2×10^{-5} / 1×10^{-4}
Weight decay / Epochs	0.01 / 12
Warmup / Schedule	0.1 / Cosine
Tversky α / β	0.3 / 0.7
SCR λ / SCL γ	0.15 / 1.5
SCR schedule	On from epoch 1
CMA gate / hidden	0.3 / 64
Scorer dims / Dropout	512, 256 / 0.1
Folds	5
Threshold τ (tuned on full)	0.45–0.50
Precision	FP16

Table 6: Full hyperparameter specification.

Marker	Count	Wt.	Avg Len	Max W
Action	4,841	1.00	26.7 ch	12
Actor	6,416	0.85	13.0 ch	6
Effect	3,739	1.10	35.4 ch	14
Evidence	3,654	1.12	35.1 ch	14
Victim	3,315	1.20	12.5 ch	6

Table 7: Training set statistics per marker type.

B Theoretical Analysis of SCR

Proposition 1 in the main text shows that Tversky loss admits diffuse minimizers. Here we provide two further results. The superscript (m) is dropped for clarity.

Proposition 2 (SCR as total variation constraint)**.** *The soft boundary count* $\hat{B} = \frac{1}{2}TV(\mathbf{p})$ *where* $TV(\mathbf{p}) = \sum_{i=1}^{N-1} |p_{i+1} - p_i|$ *is the discrete total variation. The SCR penalty* $(\hat{B} - B^*)^2$ *is therefore equivalent to:*

$$\mathcal{L}_{SCR} = \left(\frac{TV(\mathbf{p})}{2} - B^* \right)^2 \tag{9}$$

This constrains the total variation to be close to $2B^*$*, the TV of any binary sequence with* B^* *contiguous spans. The diffuse solution from Proposition 1 has* $TV = 0$ *(constant* $\mathbf{p}$*), so* $\mathcal{L}_{SCR} = (B^*)^2 > 0$ *whenever gold spans exist, penalizing the diffuse minimizer.*

Proposition 3 ($\hat{B}$ is a consistent underestimate)**.** *For any* $\mathbf{p} \in [0,1]^N$*, let* $\hat{\mathbf{y}} = \mathbf{1}[\mathbf{p} \geq \tau]$ *be the thresholded prediction. Then* $\hat{B}(\mathbf{p}) \leq \hat{B}(\hat{\mathbf{y}})$*, with equality when* $p_i \in \{0,1\}$ *for all* i*.*

Proof. For binary $\hat{\mathbf{y}}$, each transition $|y_{i+1} - y_i| \in \{0,1\}$. For soft $\mathbf{p}$, if p_i transitions gradually from below τ to above τ over multiple positions, $\sum |p_{i+1} - p_i|$ across the transition sums to $|p_{\text{high}} - p_{\text{low}}| \leq 1$, which equals the single binary transition. Intermediate values cannot increase TV beyond the binary case. □ □

This means SCR with target B^* pushes $\hat{B}(\mathbf{p})$ upward toward B^*, which (by the consistency proposition) requires $\mathbf{p}$ to have sharper transitions, approaching the binary span structure needed for threshold decoding.

C Distribution Statistics Across Splits

The 0.16-point gap between dev (0.40) and test (0.24) Macro F1 prompted us to compare per-split statistics. The dev set is a 100-sample slice of the same overall distribution, but at small n the marker mix and topical mix vary noticeably (Table 8). Average span length on dev (29.7 characters) is longer than on train (23.4), and the type distribution differs as well: Effect and Evidence together make up 31% of dev gold spans vs 34% of train, while Victim is over-represented on dev (16% vs 15%). Topically, the dev set is more diverse than train: top dev subreddits (`conspiracy`, `stupidpol`, `Christianity`, `Africa`) include several with no equivalent train mass. Without test-set gold labels we cannot directly attribute the gap to a specific shift.

Statistic	Train	Dev
n_{docs}	4,316	100
mean chars	418.0	421.6
mean words	69.9	68.7
mean spans/doc	5.1	4.6
mean span length	23.4	29.7
Per-type span counts		
Action	4,841	104
Actor	6,416	136
Effect	3,739	71
Evidence	3,654	73
Victim	3,315	72

Table 8: Train vs. dev distribution statistics.

D Error Taxonomy and Type Confusion

For each of the 456 dev gold spans, we classified the corresponding ensemble prediction into one of: Correct (IoU ≥ 0.5 and matching type), Partial (IoU $\in (0, 0.5)$ and matching type), Confused (IoU ≥ 0.3 but mismatched type), Missed (IoU < 0.3 for any prediction), or Hallucinated (predictions with no matching gold span; counted per-type). Table 9 summarizes the result. Action exhibits the highest hallucination rate, consistent with its recall-precision profile (Table 11). Confusion is heavily asymmetric: 37 gold Effect spans are matched by predicted Actions, but only 6 gold Actions are matched by predicted Effects.

Type confusion matrix. For each of the 456 gold spans, we also recorded which type the model predicted at the same character offsets (using greedy IoU matching across all predicted spans of any type, IoU ≥ 0.3). Table 10 reports the resulting

Type	Correct	Partial	Confused	Missed	Halluc.
Action	22	39	3	40	110
Actor	59	12	19	46	41
Effect	10	15	32	14	35
Evidence	10	13	24	26	32
Victim	24	5	17	26	20
Total	125	84	95	152	238

Table 9: Error taxonomy on the 100-sample dev set, full ensemble. "Correct" requires IoU ≥ 0.5 *and* type match; "Confused" requires IoU ≥ 0.3 but a type mismatch.

Gold-vs-Predicted matrix. The CMA's effective learned cross-type dependencies, read off as the dominant off-diagonal entries, are: **Effect** $\rightarrow$ **Action** (37), **Evidence** $\rightarrow$ **Action** (16), **Actor** $\rightarrow$ **Action** (11), and **Victim** $\rightarrow$ **Action** (11). The reverse direction is markedly weaker (Action $\rightarrow$ Effect: 6; Action $\rightarrow$ Evidence: 4), confirming that the learned coupling is asymmetric: predictions of long-span causal types (Effect, Evidence) are absorbed into the broader Action category, while short noun-phrase types (Actor, Victim) are largely diagonal. We treat this as a proxy for the learned correlation matrix $\mathbf{C}$, since $\mathbf{C}$ acts on per-token sigmoid probabilities and its effect is most readable through downstream prediction co-occurrence patterns.

Gold \ Pred	Act.	Actr.	Eff.	Evi.	Vic.	none
Action	**52**	1	6	4	1	40
Actor	11	**70**	3	5	1	46
Effect	*37*	0	**17**	3	0	14
Evidence	*16*	7	1	**22**	1	26
Victim	*11*	3	3	1	**28**	26

Table 10: Type confusion on dev. Rows are gold span types; columns are predicted types (or "none" if no prediction matched at IoU ≥ 0.3). Diagonal in **bold**; the dominant off-diagonal absorption into Action shown in *italics*.

Per-document Macro F1 ranges from 0 to 1 (mean 0.38, median 0.36, std 0.28); 21 of 100 documents have F1=0 and 7 of 100 have F1$\geq$0.8. Per-document F1 correlates most strongly with the number of distinct marker types present (Pearson $r = 0.51$, $p < 10^{-4}$) and the overall marker density (Pearson $r = 0.47$), suggesting the model fares better on richly annotated documents than on sparsely annotated ones.

E Per-type Error Analysis

We examined dev predictions to identify systematic patterns. Table 11 summarizes per-type error

counts and boundary precision for the full system.

	P	R	TP	FP	$\lvert\Delta_s\rvert$	$\lvert\Delta_e\rvert$
Action	.24	.56	58	179	20.7	37.9
Actor	.58	.52	71	52	3.1	1.9
Effect	.30	.38	27	62	26.0	15.8
Evidence	.33	.30	22	44	8.1	15.7
Victim	.49	.43	31	32	3.2	2.2

Table 11: Per-type error analysis on dev. $|\Delta_s|, |\Delta_e|$: mean absolute boundary error in characters for start and end of matched spans.

Action over-prediction and boundary asymmetry. Action has the highest recall (0.56) but lowest precision (0.24), with 179 false positives. Its boundary errors are the largest: predicted start positions deviate by 20.7 characters on average and end positions by 37.9 characters, indicating the model over-extends Action spans into surrounding context. The asymmetry (end errors nearly double start errors) suggests the model correctly identifies where Actions begin but struggles to determine where they end, often including subsequent clauses that describe Effects rather than Actions.

Actor and Victim have tight boundaries. These short noun-phrase markers achieve boundary errors of only 2–3 characters on average, consistent with their constrained linguistic form. Actor achieves the highest F1 (0.55) and is least sensitive to ablation changes. The small boundary errors suggest that when the model detects these markers, it identifies their extent accurately.

Evidence and Effect are hardest. These types have the lowest F1 across all configurations. Evidence spans have moderate boundary errors (8–16 chars) but low recall (0.30), suggesting the model misses many Evidence spans entirely rather than detecting them imprecisely. Effect spans have larger boundary errors (26 chars start, 16 chars end), likely due to semantic overlap with Action spans where the boundary between an action and its consequence is ambiguous.